République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abderrahmane Mira de Béjaïa
Faculté des Sciences Exactes
Département d'Informatique

Mémoire de Fin de Cycle

En Vue de l'Obtention du Diplôme Master Professionel en Informatique Génerale
Spécialité : *Administration et Sécurite des Réseaux (A.S.R)*

Thème

Sécurité du routage et d'acheminement des données dans les réseaux mobiles ad hoc

Presenté par :
Mr. CHIKER Mehdi
Mr. CHOULAK Elhadi

Soutenu devant le jury composé de :

Présidente :	**Dr.YESSAD Samira**
Rapporteurs :	**Pr.BOUALLOUCHE-MEDJKOUNE Louiza**
	Dr.BOUNOUNI Mahdi
Examinateurs :	**Dr.ATMANI Mouloud**
	Mme.LAHLAH Souad

Université de Béjaia, Promotion : 2016/2017

Remerciements

*N*ous aimerions remercier avant tout, **Dieu** Clément et Miséricordieux le Tout-Puissant, de nous avoir donné la force et la puissance pour pouvoir mener ce travail à terme.

*N*ous tenons à remercier très sincèrement le **PR.BOUALLOUCHE-MEDJKOUNE Louiza**, notre promotrice ainsi que le **Dr.BOUNOUNI Mahdi** co-promoteur, un tres grand merci pour vos conseils et nos discussion qui nous ont permis de recadrer ce travail et de relever bien des défis.

*N*ous remercions chaleureusement **Mme.YESSAD Samira**, qui nous a fait l'honneur de présider ce jury.

*N*ous adressons nos vifs remerciements à **M.ATMANI Mouloud** et **Mme.LAHLAH Souad**, pour avoir accepté de juger ce travail.

*E*nfin, merci à tout ceux qui ont contribué de prés ou de loin à la réalisation de ce mémoire.

Dédicace

Nous dédicaçons ce modeste travail :

À nos familles pour leur soutien tout au long de notre parcours universitaire : que ce travail soit l'accomplissement de vos voeux tant allégués, et le fruit de votre soutien infaillible.

À notre cher frere Mourad.

À nos cousins et cousines.

À nos chers amis.

Et à tous ceux que nous avons oublié de citer.

M. ***CHIKER Mehdi*** & ***CHOULAK Elhadi***

Table des matières

Liste des figures

Liste des tableaux

Notations & Abréviations

AODV : Ad hoc On-Demand Distance Vector.

ARAN : Autheticated Routing for Ad hoc Network.

CCS : Crédit Clearance Service.

CS : Contribution System.

DBF : Distributed Bellman-Ford.

DSDV : Destination-Sequenced Distance Vector.

DSR : Dynamic Source Routing.

DPRAODV : Detection, Prevention and Reactive AODV.

DSA : Digital Signature Algorithm.

IAACK : Improved AACK.

IP : Internet Protocol.

MAC : Media Access Control.

MANET : Mobile Ad Hoc Network.

NAM : Network Animator.

OLSR : Optimized Link State Routing.

PDA : Personal Digital Assistant.

RREP : Route Reply.

RREQ : Route Request.

SN : Selfish Node.

S-ACK : Secure-ACK.

ZRP : Zone Routing Protocol.

TCL : Tool Command Language.

TCP : Transmission Control Protocol.

UDP : User Datagram Protocol.

VANET : Vehicular Ad-Hoc Network.

WSN : Wireless Sensor Network.

ZRP : Zone Routing Protocol.

Introduction générale

Un réseau mobile ad hoc (MANET : Mobile Ad hoc Network) est une collection de nœuds qui peuvent communiquer entre eux sans compter sur une administration centralisée ou une infrastructure existante. Dans un réseau mobile ad hoc, un nœud peut communiquer directement (mode point-à-point) avec n'importe quel nœud, s'il est situé dans sa zone de transmission. Tandis que la communication avec un nœud situé en dehors de sa zone de transmission s'effectue en se basant sur la coopération des nœuds intermédiaires (communication multi-sauts). Plusieurs protocoles de routage ont été proposés pour établir la communication entre les nœuds du réseau. La plupart de ces protocoles se base sur l'hypothèse que tous les nœuds sont disposés à coopérer en se basant sur le fait que dans ce type de réseau, chaque nœud joue le rôle d'un hôte et d'un routeur. La coopération dans un protocole de routage signifie qu'un nœud achemine correctement tous les paquets destinés à être acheminés. Cependant, une telle coopération ne peut être assurée à cause des caractéristiques spécifiques de ces réseaux telle que l'absence d'une entité centrale de gestion et les ressources limitées des nœuds. Toute fois, un nœud peut refuser de coopérer avec les autres. Il peut supprimer tous les paquets destinés à être acheminés, soit pour dysfonctionner l'activité de transmission des paquets (on parle de la malveillance) ou bien pour préserver ces ressources (on parle de l'égoïsme). Un tel comportement est connu sous le nom de l'attaque de suppression de paquets.

Afin de contrecarrer l'attaque de suppression de paquets, plusieurs approches ont été proposées. Ces approches peuvent être classées en deux catégories : les approches de stimulation et les approches de réputation. Les approches de stimulation visent à motiver la coopération des nœuds. La plupart de ces approches utilisent des crédits comme moyen d'incitation. Donc, un nœud gagne des crédits en acheminant les paquets des autres. Ensuite, il utilise ces crédits pour faire acheminer ses propres paquets. Les approches de réputation visent à déterminer si un nœud est coopératif compte tenu de sa valeur de réputation. La valeur de réputation d'un nœud est calculée en surveillant son comportement dans le processus de transmission des paquets de données. Donc, suivant la technique de surveillance utilisée,

on distingue deux types d'approches : les approches basées sur le mode promiscuous et les approches basées sur l'acquittement des messages. Les approches en mode promiscuous se basent sur l'écoute des transmission des nœuds voisins. Bien que ces approches permettent d'identifier les nœuds malveillants, elles présentent plusieurs limitations liées à l'utilisation de l'écoute. Pour remédier à ces limitations, les approches en mode acquittement ont été proposées. Ces approches se basent sur l'envoi des paquets d'acquittements afin de détecter les liens malveillants.

Bien que les approches basées sur l'acquittement permettent de faire face à plusieurs limitations des approches basées sur le mode promiscuous, elles présentent aussi plusieurs limitations qui peuvent influencer leurs performances. Les principales limitations sont les suivantes :

- La détection des liens malveillants au lieu des nœuds malveillants. Cette limitation offre aux nœuds malveillants plus de chance de supprimer plus de paquets de données.
- Les nœuds en collusion peuvent contourner ces approches facilement. Deux nœuds sont en collusion, signifie qu'ils collaborent entre eux afin de cacher leurs comportements malveillants.
- L'absence d'un mécanisme de stimulation. Ces approches ne permettent pas de motiver la coopération des nœuds égoïstes.

Dans notre mémoire, nous nous sommes focalisés sur les approches basées sur l'acquittement. Pour contrecarrer leurs limitations, nous avons proposé une nouvelle approche, dénommée IAACK (Improved AACK). L'approche IAACK vise à détecter et exclure les nœuds malveillants, tout en motivant l'acheminement des paquets par les nœuds égoïstes. L'approche IAACK est structurée autour de trois composants. Le composant de surveillance est responsable de détecter des éventuelles suppressions de paquets, ou bien des collusions entre les nœuds. Le composant de réputation est responsable d'évaluer la réputation des nœuds et d'isoler les nœuds ayant des valeurs de réputation inférieures au seuil prédifini. Le composant de stimulation motive la coopération des nœuds dans le processus de découverte de routes via l'utilisation des crédits.

Structure du Mémoire

Notre mémoire est structuré en quatre chapitres :

Dans **le premier chapitre**, nous présentons des généralités sur les réseaux mobiles ad hoc à savoir leurs caractéristiques, leurs domaines d'application et les différents protocoles de routage.

Dans **le deuxième chapitre**, nous dressons un état de l'art sur l'attaque de suppression de paquets ainsi que les approches proposées pour contrecarrer cette attaque.

Dans **le troisième chapitre**, nous présentons les détails de notre approche IAACK (Improved AACK).

Le quatrième chapitre est consacré à la simulation de l'approche proposée, dont les résultats de simulation ont été discutés.

Chapitre 1

Généralités sur les réseaux mobiles ad-hoc

1.1 Introduction

Les techniques de communication progressent d'une manière vertigineuse depuis quelques années, toujours plus efficaces, elles deviennent incontournables dans tous les domaines ; Services informatiques, télécoms, messagerie, internet. Les réseaux sans fil occupent la grande part de ces techniques de communication et peuvent être classées selon le mode de communication en deux grandes catégories :

- **les réseaux sans fil avec infrastructure** : Dans cette catégorie, le réseau repose sur l'utilisation d'une infrastructure (Point d'accès ou station de base) afin de gérer l'échange d'informations entre les différents appareils mobiles du réseau.
- **Les réseaux sans fil sans infrastructure (Ad-hoc)** : Dans cette catégorie, la communication entre les appareils mobiles ne repose sur aucune infrastructure car elle se fait directement entre les différents appareils. Ce type de communication est appelé communication Ad Hoc.

1.2 Réseaux mobiles Ad Hoc : généralités

Les réseaux mobiles ad hoc [1], aussi appelés MANET (Mobile ad hoc Network), sont des réseaux distribués et auto-organisés. Il s'agit d'une collection de nœuds mobiles capables de communiquer sans compter sur une infrastructure existante ou une administration centralisée. La figure 1.1 montre une topologie classique d'un réseau ad hoc. Comme le montre cette figure, les nœuds ne s'appuient pas sur un point d'accès ou une station de base pour communiquer entre eux. Par contre, les nœuds s'appuient sur leurs voisins pour communiquer entre eux.

Les nœuds commencent par découvrir leur voisinage (les nœuds qui sont à l'intérieur de leur zone de couverture). Ensuite, à chaque fois que l'un d'entre eux souhaite communiquer avec un nœud distant, il envoie son message à ses voisins qui à leur tour l'envoient au plus proche voisin (suivant la route utilisée) et ainsi de suite. Ce processus est répété jusqu'à ce que le message atteigne le nœud destinataire.

Les nœuds des réseaux ad hoc peuvent être de natures très diverses (ordinateur, PDA, caméra sans fil, capteur, téléphone mobile, lecteur MP3, console de jeux, etc.). Ils peuvent communiquer en utilisant des technologies sans fil tels que : Bluetooth, Wifi, HiperLAN2 et UWB [2, 3].

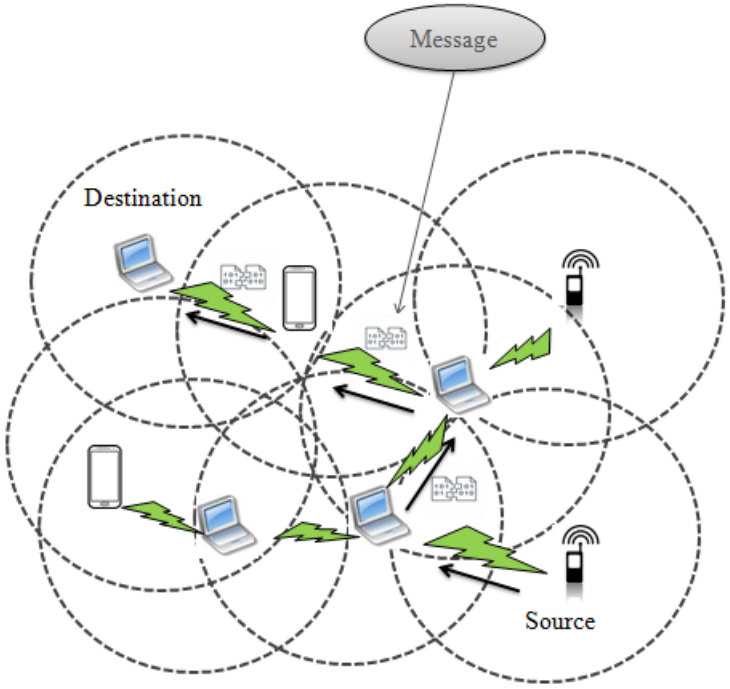


FIGURE 1.1 – Architecture d'un réseau ad hoc

1.3 Caractéristiques des réseaux mobiles Ad Hoc

Les réseaux mobiles Ad Hoc sont caractérisés par [1, 4] :

- **Topologie dynamique** : Les nœuds se déplacent indépendamment les uns des autres et de manière imprévisible. Un ou plusieurs nœuds peuvent intégrer ou sortir du réseau à tout moment .
- **Ressources limitées** : les sources d'énergie telles que les batteries sont nécessaires pour la communication des nœuds mobiles. Malheureusement, ces sources d'énergie ont

une durée de vie limitée et leur épuisement dépend des traitements effectués au niveau du nœud (les opérations de transmission, réception et les calculs complexes, etc). Par conséquent, la consommation d'énergie est l'une des contraintes de ces réseaux. La gestion de l'énergie est nécessaire pour les nœuds dans le but de conserver et d'augmenter leur durée de vie (par conséquent la durée de vie du réseau).

- **Sécurité faible** : Due à l'absence d'une entité centrale de gestion, les problèmes de sécurité sont plus complexes que pour un réseau classique. Ayant pour cause la nature des liaisons radio (diffusion), la sécurité des réseaux ad hoc est difficile à assurer. Le réseau peut être vulnérable à plusieurs attaques, telles que l'écoute, le spoofing et le déni de service.
- **L'absence d'infrastructure** : Les réseaux ad hoc se distinguent des autres réseaux mobiles par l'absence d'infrastructures préexistantes et de tous genres d'administration centralisée. Les hôtes mobiles sont responsables d'établir et de maintenir la connectivité du réseau de manière continue.

1.4 Domaines d'applications des réseaux mobiles Ad hoc

Leurs facilité de mise en place, absence d'infrastructure et coût réduit élargissent le domaine d'application des réseaux mobiles Ad Hoc [4, 5, 6]. Parmi ces applications, on peut citer :

- **Applications militaires :** Les réseaux mobiles ad hoc sont conçus à la base pour des applications et des opérations à caractère militaire. Ces réseaux sont adaptés aux environnements hostiles, car ils sont dynamiques et rapidement déployables. Les nœuds du réseau ne sont que des équipements militaires communicants (soldats, véhicules blindés, etc.). Cependant, l'utilisation de ces réseaux a dépassé le domaine militaire grâce au développement technologique des réseaux sans fil.
- **Opérations de sauvetage :** Les réseaux mobiles ad hoc sont aussi utilisés lors des opérations de sauvetage, notamment lors de tremblements de terre ou autres catastrophes. Ils peuvent être rapidement déployés sur des terrains de sinistres pour assurer le relais et la liaison des communications entre sauveteurs.
- **Réseaux de capteurs sans fil(WSN) :** Un réseau de capteurs sans fil est un réseau ad hoc avec un grand nombre de nœuds qui sont des micro capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils peuvent être

aléatoirement dispersés dans une zone géographique, appelée « champ de captage » correspondant au terrain d'intérêt pour le phénomène capté.

– **Réseau d'entreprises :** : La facilité à déployer ces réseaux et leur coût réduit intéressent de plus en plus les entreprises. Cela permet d'assurer une grande mobilité des agents, le partage des données. Par exemple, lors d'une réunion ou conférence, l'intervenant peut communiquer avec tous les participants et créer un débat interactif.
– **Réseaux véhiculaires VANET :** Les VANET sont considérés comme étant un cas particulier des réseaux mobiles ad hoc [5]. Ils se constituent à partir d'un ensemble d'entités communicantes, composées de véhicules et d'unités de bords de route (RSU). Grâce aux différentes applications que supportent les VANET, ces réseaux sont considérés comme étant le moyen le moins cher pour éviter les embouteillages, minimiser la consommation de carburant et réduire le temps passé sur les routes.

1.5 Le routage dans les réseaux mobiles ad hoc

1.5.1 Définition du routage

Le routage est une méthode à travers laquelle on fait transiter une information donnée depuis un certain émetteur vers un destinataire bien précis. Le problème du routage ne se résume pas seulement à trouver un chemin entre les deux nœuds du réseau, mais encore à trouver un acheminement optimal et de qualité pour son transit.

1.5.2 Classification des protocoles de routage

Le principal but de toute stratégie de routage est de mettre en œuvre une bonne gestion d'acheminement. Elle doit être robuste, efficace et sûre. Les protocoles de routage sont de manière générale classés en trois grandes catégories :

- Protocoles de routage proactifs.
- Protocoles de routage réactifs.
- Protocoles de routage hybrides.

1.5.3 Les protocoles proactifs

Dans cette classe, chaque nœud doit avoir des tables de routage vers toute destination atteignable. Pour atteindre cet objectif en permanence, le nœud envoie des messages de contrôle régulier même quand ces routes ne sont pas utilisées. Cela garantit la disponibilité d'une route vers toute destination demandée instantanément. Cette catégorie des protocoles de routage se basent sur deux principales méthodes qui sont :

- Méthode état de lien (link state).
- Méthode vecteur de distance (distance vector).

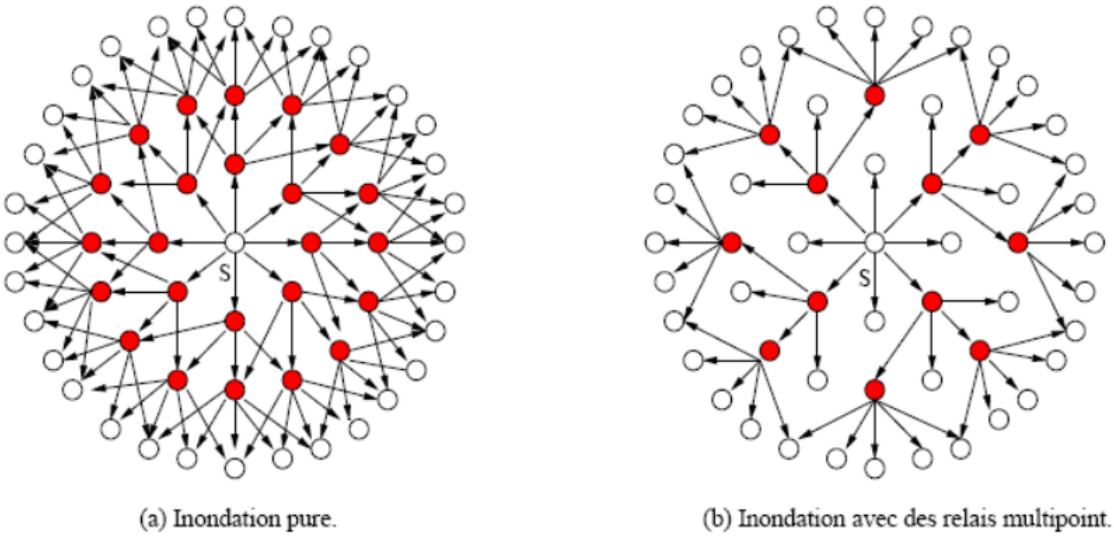
Les protocoles de routage de cette catégorie exigent une mise à jour périodique des données de routage qui est assurée par la diffusion périodique des messages de contrôle par tous les nœuds. Ils utilisent la technique du plus court chemin pour choisir un chemin parmi plusieurs. En général, la métrique utilisée pour choisir le plus court chemin est le nombre de sauts. Dans ce qui suit, nous allons présenter deux exemples de protocoles de cette catégorie : DSDV et OLSR.

DSDV

Le protocole de routage DSDV (Dynamics destination-Sequenced Distance Vector) [6, 7] se base sur l'algorithme distribué de Bellman-Ford (DBF), qui utilise les vecteurs de distance. Chaque station maintient une table de routage contenant toutes les destinations qu'elle peut atteindre, le coût (en nombre de sauts) pour atteindre chaque destination et un numéro de séquence lié au nœud destination. Cette table de routage permet à un nœud d'avoir une vision complète de la topologie du réseau. Chaque nœud diffuse périodiquement sa table de routage à ses voisins directs (un saut). Cette diffusion soit elle se fait de manière incrémentale (seules les données qui ont changé par rapport à la dernière mise à jour), ou bien de manière intégrale (la table toute entière). Dans DSDV, les numéros de séquences sont employés pour déterminer la fraîcheur des informations de routage. Donc, la route avec le numéro de séquence le plus élevé est choisie (information fraîche).

OLSR

Le protocole de routage OLSR (Optimized Link State Routing) est un protocole par état de lien qui utilise une technique optimisée pour la diffusion des informations de contrôle. La solution consiste à ne permettre qu'à un sous-ensemble de voisins de retransmettre les messages (voir figure 1. 2) [8]. Ces voisins sont appelés les relais multipoints ou MPRs (multipoint relayes). Chaque nœud effectue la sélection de ses MPRs en se basant sur le nombre de voisins à deux sauts atteignables via ce nœud . Seuls les nœuds MPR sont responsables d'assurer les tâches de diffusion des informations de contrôle.



(a) Inondation pure. (b) Inondation avec des relais multipoint.

FIGURE 1.2 – Diffusion par inondation et diffusion optimisée dans OLSR

1.5.4 Les protocoles Réactifs

Comme nous avons vu dans la section précédente, les protocoles de routage proactifs essaient de maintenir les meilleures routes vers toutes les destinations possibles. Les routes sont sauvegardées même si elles ne sont pas utilisées. La sauvegarde permanente des routes est assurée par un échange continu des informations de routage. Cependant, les protocoles de routage réactifs (à la demande) créent et maintiennent les routes selon les besoins. Lorsqu'un nœud a besoin d'une route, une procédure de découverte de routes est lancée. Cette procédure s'achève par la découverte d'une ou plusieurs routes à la destination désirée. La validité des routes trouvées est assurée par le processus de maintenance de routes.

Nous allons présenter quelques exemples de protocoles de routage réactifs : AODV et DSR.

DSR

Le protocole DSR (Dynamic Source Routing) [9] se décompose en deux processus complémentaires : le processus de découverte de routes et le processus de maintenance de routes. Dans le processus de découverte de routes, la source diffuse une requête RREQ (qui contient un champs d'enregistrement de la route traversée) dans son voisinage. Chaque voisin recevant cette RREQ insère son identifiant dans la RREQ et la rediffuse à son tour dans son voisinage (voir la figure 1.3). Et ce en complétant le champ d'enregistrement de routes de leur propre identifiant, jusqu'à atteindre la destination cible. Une fois que la RREQ atteint

la destination, le nœud destinataire répond à la source en envoyant un paquet réponse de route RREP en utilisant la route inversée incluse dans le paquet RREQ. Le processus de maintenance de routes est utilisé pour assurer la validité de la route utilisée en y identifiant et évitant les liens défaillants. Si un nœud détecte un lien défaillant, il invalide toutes les routes impliquant ce nœud dans son cache de route. Ensuite, il envoie un paquet erreur de route RERR (Route Error) à la source en y identifiant le lien défaillant. Une fois que la source reçoit le paquet RERR, elle invalide toutes les routes impliquant ce lien. La source cherche dans son cache de route si elle possède une autre route valide à la destination, si c'est le cas, elle l'utilise directement. Autrement, elle lance le processus de découverte de routes à nouveau.

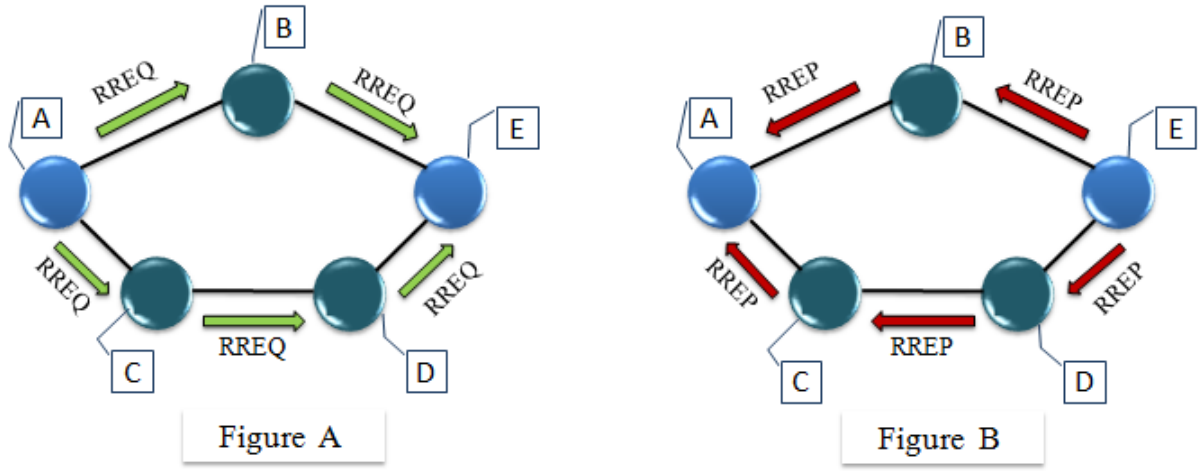


FIGURE 1.3 – Processus de découverte de routes

AODV

Le protocole AODV [10] (Ad hoc On Demand Distance Vector), représente essentiellement une amélioration de l'algorithme proactif DSDV. Le protocole AODV réduit le nombre de diffusions de messages. Cela en créant les routes lors du besoin, contrairement au DSDV, qui maintient la totalité des routes. Le protocole AODV est fondé sur l'utilisation des deux mécanismes « Découvertes de route » et « maintenance de route », en plus du routage nœud par nœud, le principe des numéros de séquence et l'échange périodique du DSDV.

AODV utilise les principes des numéros de séquence afin de maintenir la consistance des informations de routage. A cause de la mobilité des nœuds dans les réseaux ad hoc, les routes changent fréquemment, ce qui fait que les routes maintenues, par certains nœuds, deviennent

invalides. Les numéros de séquence permettent d'utiliser les routes les plus nouvelles ou autrement dit les plus fraîches (Fresh routes). Ce dernier utilise une requête de route dans le but de créer un chemin vers une certaine destination.

1.5.5 Les protocoles hybrides

Les protocoles de routage hybrides sont proposés en tirant profit des avantages des protocoles de routage proactifs et réactifs. Les nœuds du réseau sont regroupés en zone. L'approche de routage proactive est utilisée pour faire communiquer les nœuds d'une zone. Tandis la communication entre les nœuds distants (hors zone) est réalisée en utilisant l'approche de routage réactive. Le protocole ZRP (Zone Routing Protocol) [11] est un exemple de protocole de routage hybride. Dans ZRP, chaque nœud maintient une table de routage pour sa zone. Les informations de routage sont régulièrement diffusées par tous les nœuds d'une zone (routage proactif dans cette zone). Pour atteindre tout autre nœud qui n'apparaîtrait pas dans sa zone (communication inter-zone), un nœud a recours à un protocole de routage de type réactif. A noter qu'une zone regroupe tous les nœuds qui sont à une certaine distance (en terme de nombre de sauts).

1.6 Conclusion

Dans ce chapitre, nous avons présenté quelques notions de base des réseaux mobiles ad hoc, à savoir leurs caractéristiques les plus importantes, leurs domaines d'applications ainsi que quelques protocoles de routage de base utilisés pour établir la communication entre les nœuds du réseau. Dans le chapitre suivant, nous aborderons la problématique de sécurité des réseaux mobiles ad hoc à savoir les différentes vulnérabilités, les attaques et mécanismes de sécurité existants.

Chapitre 2

État de l'art sur l'attaque de suppression de paquets

Introduction

Dans ce chapitre, nous allons présenter les différents aspects de la sécurité dans les réseaux mobiles ad hoc. On se focalisera principalement sur l'attaque de suppression de paquets et les différentes approches proposées pour la contrecarrer .

2.1 Besoins en sécurité

Les besoins de sécurité des réseaux mobiles Ad-hoc [4] sont plus ou moins les mêmes que ceux des réseaux filaires ou sans fil avec infrastructure. La sécurisation de ces réseaux est basée sur cinq concepts fondamentaux à savoir :

2.1.1 Authentification

Elle permet de vérifier l'identité d'un nœud dans le réseau. C'est une étape incontournable pour le contrôle de l'accès aux ressources du réseau. Sans elle, un nœud malicieux peut facilement usurper l'identité d'un autre nœud dans le but de bénéficier de ses privilèges, ou d'effectuer des attaques sous son identité.

2.1.2 Confidentialité

Les informations ne sont accessibles que pour les nœuds autorisés. Si la confidentialité n'est pas assurée, un nœud malveillant pourrait accéder aux informations secrètes contenues dans les messages transitant par lui.

2.1.3 Intégrité

Ce service assure que le trafic de la source à la destination n'a pas été altéré ou modifié sans autorisation préalable pendant sa transmission. C'est la protection contre les menaces qui peuvent altérer la configuration du système ou des données.

2.1.4 Non-répudiation

Elle garantie que chaque nœud du réseau ne peut nier l'envoi ou la réception d'un message. Sans ce service, il sera difficile de vérifier que l'émetteur et le destinataire sont bien les parties qui disent avoir respectivement envoyé ou reçu le message.

2.1.5 Disponibilité

La disponibilité consiste à garantir la continuité du service fourni par un nœud même en présence d'une attaque.

2.2 Classification des attaques dans les réseaux mobiles ad hoc

2.2.1 Attaque interne Vs attaque externe

Si l'attaquant ou le nœud malicieux se trouve dans le réseau, on parle d'une attaque interne. Dans le cas où le il se connecte depuis l'extérieur, on dira que l'attaque est externe.

2.2.2 Attaque individuelle Vs attaque en collusion

Dans l'attaque individuelle [12], un seul nœud lance l'attaque. Il existe des attaques lancées par plusieurs nœuds, c'est se qu'on appelle attaque en collusion (les nœuds malveillants collaborent entre eux). Ce genre d'attaques est plus dangereux et difficile à détecter.

2.2.3 Attaque passive Vs attaque active

Dans l'attaque passive [13], l'attaquant intercepte et analyse le trafic pour déterminer les relations entre les nœuds et éventuellement déterminer les nœuds importants dans le fonctionnement du réseau. Tandis dans l'attaque active, elle vise à perturber le fonctionnement du réseau en supprimant, modifiant, fabriquant des paquets ou en rejouant d'anciens paquets. Un attaquant peut lancer une attaque Blackhole, Grayhole,...etc.

2.3 Attaques contre les protocoles de routage

Dans cette partie, nous présentons quelques attaques actives ciblant les protocoles de routage.

2.3.1 Attaque du trou noir (Black Hole Attack)

Dans cette attaque, un nœud malveillant falsifie les informations de routage pour forcer le passage des paquets de données par lui- même [14]. Sa seule mission est ensuite de ne rien transférer, créant ainsi une sorte de trou noir dans le réseau. Le nœud malveillant peut aussi se placer sur un endroit stratégique du routage dans le réseau. Ensuite, il supprime tous les messages destinés à être acheminés, ce qui peut causer le dysfonctionnement du réseau.

2.3.2 Attaque Gray Hole Attack

Cette attaque est similaire à l'attaque Black Hole sauf qu'au lieu de supprimer tous les paquets destinés à être acheminés, l'attaque ne supprime qu'une fraction des paquets.

2.3.3 Attaque de Trou ver (Wormhole Attack)

Cette attaque nécessite la présence d'au moins deux nœuds malveillants formant une collusion[15]. Les nœuds en collusion se trouvent généralement dans des zones géographiquement séparées. Le but de cette attaque est de former un tunnel entre ces deux nœuds. Pour perturber un protocole de routage, les nœuds en collusion peuvent sauvegarder les paquets de contrôles d'une zone pour les diffuser dans une autre zone.

2.3.4 Attaque d'usurpation d'identité (Spoofing)

Un nœud malveillant peut se faire passer pour un autre nœud honnête en usurpant son identité. Il lance une attaque en prétendant d'être un autre nœud .

L'usurpation d'identité peut conduire à des boucles de routage (voir figure 2.1). Dans la (figure A), chacun des nœuds A, B, C, D possède une route vers la destination X. Pour former une boucle de routage, le nœud malveillant M change son adresse MAC pour qu'elle corresponde à celle de A, puis il annonce à B une route vers X avec une métrique meilleure (un nombre de sauts plus petits ou un numéro de séquence plus récent) que celle de la route à travers C. B met à jour alors sa table de routage en sauvegardant la nouvelle route vers X à travers A (figure B). Le nœud malveillant M répète le même processus avec le nœud C en usurpant l'identité du nœud B. C enregistre alors la nouvelle route vers X à travers B (figure

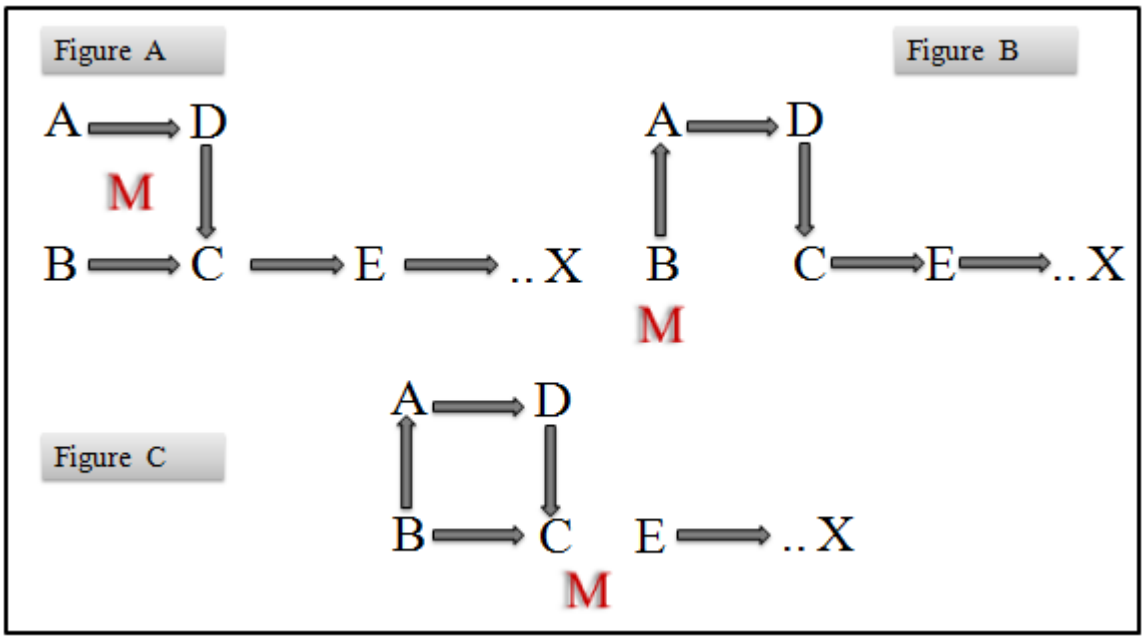


FIGURE 2.1 – Création de boucle de routage par Spoofing

C). De cette façon une boucle est formée et aucun paquet de l'un des quatre nœuds A, B, C, D ne peut arriver à X.

2.3.5 Attaque de suppression de paquets

L'attaque de suppression de paquets consiste à supprimer les paquets destinés à être acheminés. Un attaquant peut supprimer les paquets pour dysfonctionner le processus de routage ou bien pour préserver ses ressources, vu que la transmission des paquets consomme des ressources.

2.4 Description de l'attaque de suppression de paquets

L'attaque de suppression de paquets consiste à supprimer les paquets destinés à être acheminés. Elle peut être lancée par deux types d'attaquants : attaquant malveillant et attaquant égoïste. Nous prenons le protocole de routage DSR pour illustrer l'attaque de suppression de paquets.

2.4.1 Attaquant malveillant

L'attaquant malveillant supprime les paquets de données destinés à être acheminés afin de perturber le processus de transmission des paquets de données. Il peut le faire par quatre méthodes.

Méthode 1

L'attaquant lance l'attaque de trou noir (Black Hole), il répond toujours d'une manière positive à toutes les RREQ en envoyant une fausse RREP. Il prétend toujours avoir la route la plus optimale vers la destination afin d'obliger la source des paquets à utiliser cette route. Donc, tous les paquets acheminés suivant cette route seront supprimés.

Méthode 2

L'attaquant lance l'attaque Gray Hole. Cette attaque est similaire à l'attaque Black Hole, sauf que l'attaquant ne supprime qu'une partie des paquets de données passant par lui.

Méthode 3

L'attaquant participe dans le processus de découverte de route. Il achemine tous les RREQ et les RREP, sauf que lorsque il fait partie d'une route, il supprime tous les paquets passant par lui.

Méthode 4

Il utilise le même principe que la méthode 3, mais il supprime qu'une partie des paquets.

2.4.2 Attaquant égoïste

Un attaquant égoïste ne vise pas à endommager le processus de routage. Il vise à préserver ses ressources en supprimant les paquets destinés à être acheminer, car la transmission de ces derniers lui consomme des ressources. Il existe trois types de nœuds égoïstes liés au protocole DSR [16].

Nœuds égoïstes de Type 1 (SN1)

Ces nœuds participent aux phases de découverte et de maintenance de route, mais ils refusent de retransmettre les paquets de données.

Nœuds égoïstes de Type 2 (SN2)

Ces nœuds ne participent pas à la phase de découverte de route. Ils utilisent seulement leur énergie pour la transmission de leurs propres paquets.

Nœuds égoïstes de Type 3 (SN3)

Ces nœuds se comportent (ou se comporteront différemment) en fonction de leur niveau d'énergie. Lorsque leur niveau d'énergie se situe entre l'énergie maximal E et un seuil T1, le nœud se comporte correctement. Si le niveau d'énergie du nœud est entre T1 et un autre seuil inférieur T2, il se comporte comme un nœud de type SN1. Enfin, si le niveau d'énergie est inférieur à T2, il se comporte comme un nœud de type SN2. La relation entre T1, T2 et E est T2 < T1 < E (voir figure 2.2).

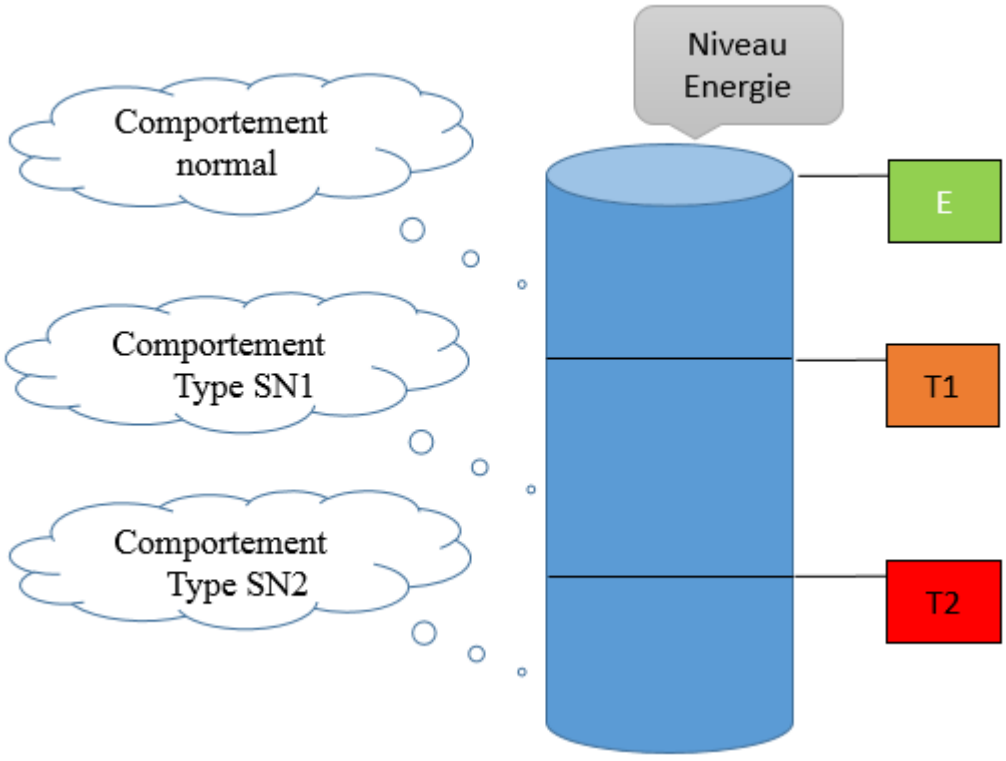


FIGURE 2.2 – Nœuds égoïstes de Type 3 (SN3)

2.5 Les approches existantes

Plusieurs approches de sécurité ont été proposées pour assurer la sécurité dans les réseaux mobiles ad hoc [17, 18, 19, 20]. La plupart de ces approches se basent sur les méthodes de cryp-

tographie, système de détection d'intrusion (IDS), PKI, et tiers de confiance (TTP). Toutes ces méthodes permettent d'assurer plusieurs propriétés de sécurité telles que l'intégrité, la confidentialité et l'authentification.

Cependant, elles ne permettent pas de contrecarrer l'attaque de suppression de paquet, car elles ne peuvent pas assurer qu'un nœud a acheminé les paquets qui sont censés être acheminés. Pour garantir cette propriété, les approches de stimulation et réputation ont été proposées.

2.5.1 Approche de stimulation

Les approches de stimulation ont été proposées dans le but de fournir des motivations pour que les nœuds coopèrent. La plupart des approches de cette catégorie emploient des crédits comme motivation pour que les nœuds acheminent les paquets. Les nœuds qui transmettent des paquets reçoivent des crédits, et en retour, ils peuvent utiliser ces crédits pour faire acheminer leurs propres paquets.

Nuglets

Buttyan et Hubaux ont proposé une approche basée sur les Nuglets [21]. Les Nuglets sont considérés comme des crédits. Un nœud gagne des Nuglets en acheminant les paquets des autres. En revanche, il perd des Nuglets pour faire acheminer ses propres paquets. Afin qu'un nœud puisse acheminer ses propres paquets, il doit maintenir son compteur Nuglets positif. L'approche Nuglets nécessite l'utilisation d'un matériel résistant à la manipulation appelé module de sécurité. Ce module est utilisé afin d'éviter qu'un nœud ne puisse modifier son compteur Nuglets de manière illégitime.

SPRITE

Dans l'approche SPRITE, les auteurs [22] ont incorporé une entité de confiance, appelée CCS (Crédit Clearance Service), qui gère tout le système de crédit. Lorsqu'un nœud reçoit un paquet, il garde une trace de la réception de ce message (sous forme de reçu ou d'accusé de réception). Lorsque ce même nœud obtient une connexion avec le CCS, il lui transmet régulièrement tous les reçus des messages qu'il a relayés. Le CCS détermine alors le crédit à donner au nœud pour la totalité de ses transmissions, ainsi que le prix que doit payer l'initiateur de chacun de ces messages.

Ocean (Observation based Cooperation Enforcement in Ad hoc Networks)

Dans l'approche OCEAN [23], chaque nœud maintient un compteur de crédit appelé chip-count pour chaque nœud voisin. En acheminant les paquets d'un nœud voisin, un nœud gagne des chips (crédits). Lorsqu'un nœud achemine les paquets d'un nœud voisin, ce dernier perd des chips chez ce nœud. Pour qu'un nœud achemine les paquets d'un nœud voisin, il doit vérifier le compteur chip-count associé à ce voisin. Si le compteur chip-count est inférieur à un seuil prédéfini T, alors le nœud voisin est puni et sa demande est rejetée.

Head (Hybrid mechanism to Enforce node cooperation in mobile AD hoc network)

Afin d'améliorer l'approche OCEAN, les auteurs dans [24] ont proposé une nouvelle approche, appelé HEAD. Cette approche consiste à conserver une table de réputation dans chaque nœud. Chaque entrée de la table correspondant à un nœud voisin et se compose de quatre entrées : ID, rating, avoid-list, non-chip-list. L'ID contient l'identifiant du nœud voisin, rating représente la réputation du nœud voisin, avoid-list et non-chip-list représentent la liste des nœuds malveillants et égoïstes détectés. Cette approche se base sur la surveillance du voisinage pour calculer les valeurs de réputation des nœuds voisins. Si cette valeur tombe en dessous d'un certain seuil, alors le nœud est considéré comme malveillant et il est rajouté à la liste avoid-list. L'utilisation des compteurs chip-count est similaire à celle de l'approche OCEAN. La différence principale est que : Si le compteur chip-count d'un nœud tombe en dessous d'un certain seuil prédéfini, le nœud est considèré comme un nœud égoïste et il est ajouté a la liste non-chip list. Afin de punir les nœuds maleveillants et les nœuds égoïstes, chaque nœud diffuse d'une manière périodique des messages contenant les deux listes. Lorsque le nœud reçoit une RREQ il vérifie si l'initiateur fait partie de ces deux listes, si c'est le cas il le punit en lui refusant d'acheminer ses paquets.

Approche [25]

Les auteurs dans [25] ont proposé une approche pour motiver les nœuds à coopérer dans le processus de découverte de route, sans utiliser la notion de crédit. Cette approche ne nécessite ni un matériel inviolable ni un serveur central. Elle se base sur la nature de transmission des paquets RREQ. Chaque nœud surveille les activités de transmission du paquet RREQ de ses voisins. Ensuite, il calcule le rapport entre le nombre de RREQ routés par lui-même et par son nœud voisin. Si le rapport est supérieur à un certain seuil prédéfini alors le nœud est considéré comme un nœud coopératif. Autrement, il sera considéré comme un nœud égoïste. Afin de punir les nœuds égoïstes, toutes les RREQ initiées par eux seront

rejetées jusqu'à qu'ils devienent coopératifs. Bien que cette approche permet d'identifier les nœuds égoïstes, elle peut être manipulée par des nœuds agissant de manière intelligente.Ces nœuds peuvent router une partie des paquets de RREQ mais ils refusent de router leurs réponses RREP de telle façon à garder le rapport supérieur au seuil.

Approche NHACK (Novel Hybrid Acknowledgement-based)

Pour remédier à la limitation de l'approche proposée dans [25], les auteurs ont proposé un système de contribution CS [26](Contribution System). En comparaison avec l'approche [25] qui ne prend en considération que les paquets RREQ lors du calcul du rapport, le système de contribution [26] de l'approche prend en considération les paquets RREQ, RREP et RERR. Ensuite, chaque nœud calcule la contribution de chaque nœud voisin dans le processus de découverte. La contribution de chaque nœud est utilisé comme une incitation de coopération. Donc, seuls les nœuds ayant des contributions supérieures au seuil prédéfini peuvent faire acheminer leurs paquets. Autrement, toutes leurs RREQ seront ignorées.

2.5.2 Approches de réputation

L'approche de réputation consiste à déterminer si un nœud est digne de confiance compte tenu de sa valeur de réputation. La valeur de réputation est une valeur numérique qui peut être définie comme la perception d'un nœud sur un autre. Elle est calculée en observant et surveillant le comportement d'un nœud dans le processus de transmission des paquets de données. Si un nœud achemine correctement un paquet de données, sa valeur de réputation est incrémentée. Autrement, elle est décrémentée. Si la valeur de réputation d'un nœud est inférieure à un seuil prédéfini, le nœud est considéré comme malveillant. Suivant l'approche de surveillance utilisée pour surveiller le comportement des nœuds voisins, on distingue deux catégories d'approches : les approches en mode promiscuous et les approches en mode acquittement.

Approches en mode promiscuous

Dans cette catégorie d'approches, afin de surveiller l'activité de transmission des paquets de données par les nœuds voisins, chaque nœud utilise le mode promiscuous [27]. Le principe de ce mode est le suivant : si un nœud A est dans la portée de transmission de son voisin B. Le nœud A peut écouter toutes les communications de son voisin B. Les approches de base de cette catégorie sont les suivantes :

Watchdog et pathrater

Les modules Watchdog et Pathrater ont été introduits par les auteurs dans [27], avec pour but d'identifier les nœuds malveillants qui acceptent de transmettre les paquets de données mais sans jamais le faire. Le Watchdog est utilisé pour surveiller le comportement des nœuds voisins en écoutant leurs transmissions. Pour y faire, chaque nœud maintient un buffer de paquets de données récemment envoyés. Si le paquet écouté existe dans le buffer, le Watchdog considère que le nœud a acheminé le paquet de données. Autrement, si le paquet de données est maintenu un certain temps prédéfini sans être acquitté, le Watchdog considère que le nœud a supprimé le paquet de données. Dans ce cas, il incrémente le compteur de paquet supprimé associé à ce nœud . Si ce compteur est supérieur à un seuil prédéfini, le nœud surveillé est considéré comme malveillant. Le module Pathrather est utilisé pour choisir la meilleure route de transmission en utilisant les informations collectées par le module Watchdog.

CONFIDANT (Cooperation of Nodes, Fair-ness In Dynamic Ad-hoc Networks)

L'approche Confidant a été proposée par les auteurs dans [28]. Elle est composée de quatre composants qui sont : le moniteur, le système de réputation, le gestionnaire de confiance et le gestionnaire de route. Le moniteur surveille les activités de transmission des nœuds voisins. Le système de réputation met à jour les valeurs de réputation des nœuds voisins. Le gestionnaire de confiance partage les valeurs de réputation des nœuds, et il envoie des messages d'alarmes si un nœud malveillant est détecté . Enfin le gestionnaire de route permet de supprimer toutes les routes impliquant des nœuds malveillants. Afin de les punir, les requêtes initiées par ces nœuds malveillants seront rejetées.

CORE (Collaborative Reputation Mechanism)

L'approche CORE a été proposée par les auteurs dans [29]. Dans CORE, un mécanisme de réputation collaborative a été incorporé dans le protocole de routage DSR afin d'imposer une coopération nodale. Trois types de réputation ont été définis : Réputation indirecte, Réputation subjective et Réputation fonctionnelle. La réputation indirecte est calculée en se basant sur les réputations partagées par les nœuds voisins. La réputation subjective est calculée en se basant sur la surveillance des activités de transmission des nœuds voisins. La réputation fonctionnelle est calculée en fusionnant la réputation subjective et indirecte au moyen d'une formule de combinaison pondérée. La réputation fonctionnelle est relative à une fonction bien précise, par exemple la fonction de transfert de paquet. En comparaison avec l'approche CONFIDANT, dans CORE, seule les observations positives seront diffusées (les alarmes ne sont pas envoyées).

Bien que ces approches permettent d'identifier les nœuds malveillants, elles présentent plusieurs limitations. Les approches se basant sur le mode promiscuous comme technique de surveillance de voisinage ne permettent pas d'identifier les nœuds malveillants dans les cas suivants[27] :

- La collusion : c'est lorsque deux nœuds malveillants successifs suivant une route de transmission collaborent entre eux pour cacher leurs comportements malveillants.
- La collision ambigüe : en présence de collision ambigüe, le watchdog ne peut pas surveiller le comportement de ses voisins. Considérons le scénario décrit dans la figure 2.3. Une collusion s'est produite au niveau du nœud A au moment de son écoute de la transmission de B. A ne peut pas distinguer si la collusion est causée par l'envoi du nœud B ou par la transmission du nœud S.
- La puissance de transmission limitée : un nœud du réseau peut limiter sa puissance de transmission afin de contourner le Watchdog. Donc, il limite sa puissance de transmission afin d'être écouté par le nœud précédent. Cependant, le signal est trop faible pour que le nœud suivant reçoive correctement le paquet.
- La suppression partielle des paquets : cette technique de surveillance ne permet pas de détecter les nœuds malveillants supprimant une fraction des paquets de données passant par eux.
- L'attaque de fausse accusation : c'est lorsque un nœud malveillant accuse à tort un autre nœud honnête d'être malveillant, mais en réalité c'est lui le véritable nœud malveillant.

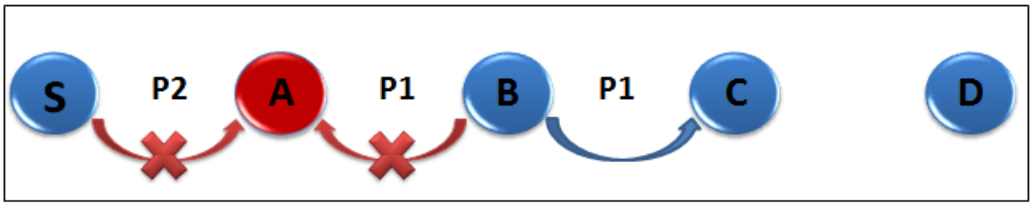


FIGURE 2.3 – Collision ambigüe

Approches basées sur les acquittements

Les approches basées sur les acquittements ont été proposées comme solution aux limitations posées dans les approches basées sur le mode promiscous. Ces approches se basent sur l'envoi des paquets d'acquittement afin de détecter les liens malveillants. Dans ce qui suit, nous présentons quelques approches de cette catégorie.

TWOACK

L'approche TWOACK a été proposée par Liu et al.[16]. Dans cette approche, les auteurs ont introduit un nouveau type de paquet d'acquittement, appelé TWOACK. Pour chaque triplet de nœud le long d'une route de transmission, le paquet TWOACK est utilisé pour assurer que le troisième nœud de la triplette a reçu correctement le paquet de donnée acheminé. Le processus de fonctionnement de l'approche TWOACK est illustré à la (figure 2.4). Soit (A,B,C) une triplette de nœud.

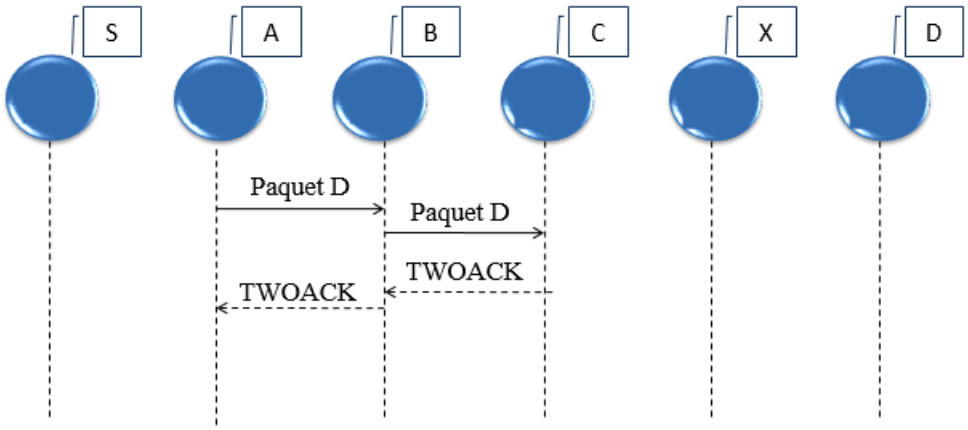


FIGURE 2.4 – Schéma TWOACK

Le nœud A transmet le paquet 1 au nœud B qui l'achemine à son tour au nœud C. Lorsque le nœud C reçoit le paquet 1, il doit générer un paquet TWOACK. Ensuite, il l'envoie au nœud A situé à deux sauts dans la direction opposée de la route de transmission. Si le nœud A reçoit un paquet TWOACK pour le paquet 1, il conclut que la transmission du paquet 1 du nœud A vers C est réussie. Ce processus est répété pour chaque triplette de nœud le long de la route. Chaque nœud maintient un compteur de paquets de données non-acquittés par chaque lien de transmission surveillée. Dans notre exemple, le nœud A maintient un compteur pour le lien B-C. Pour chaque paquet de donnée non-acquitté, le nœud A incrémente le compteur associé au lien B-C. Si ce compteur dépasse un certain seuil, le nœud A déclare que le lien B-C est malveillant. Le nœud A invalide toutes les routes impliquant ce lien et envoie un rapport de malveillance (sous forme d'une RERR) à la source pour procéder à l'isolation du lien.

2ACK

Les auteurs Iu et al [30] ont proposé un nouvelle approche appelée 2ACK. Le fonctionnement de l'approche 2ACK est semblable à l'approche TWOACK [16]. Il existe quelques différences entre ces dernierès. Tout d'abord, contrairement à l'approche TWOACK, l'approche 2ACK n'acquitte qu'une fraction des paquets de données par des paquets 2ACK au lieu de les acquitter tous. Ainsi, elle intègre un mécanisme de signature numérique qui permet d'empêcher les nœuds d'envoyer des faux paquets 2ACK.

AACK

Basée sur l'approche TWOACK, les auteurs Sheltami et al dans [31] ont proposé l'approche AACK. Dans le but de réduire le nombre de paquets TWOACK échangés entre les nœuds lorsqu'il n'y a aucune activité malveillante le long de la route de transmission. L'approche AACK est le résultat de combinaison de deux modes : ACK et TACK. Le mode ACK est similaire au mode d'acquittement de bout en bout. Le mode par défaut utilisé est le mode ACK. Dans ce mode, la destination renvoie un paquet ACK pour chaque paquet de donnée reçu, comme le montre la (figure 2.5).

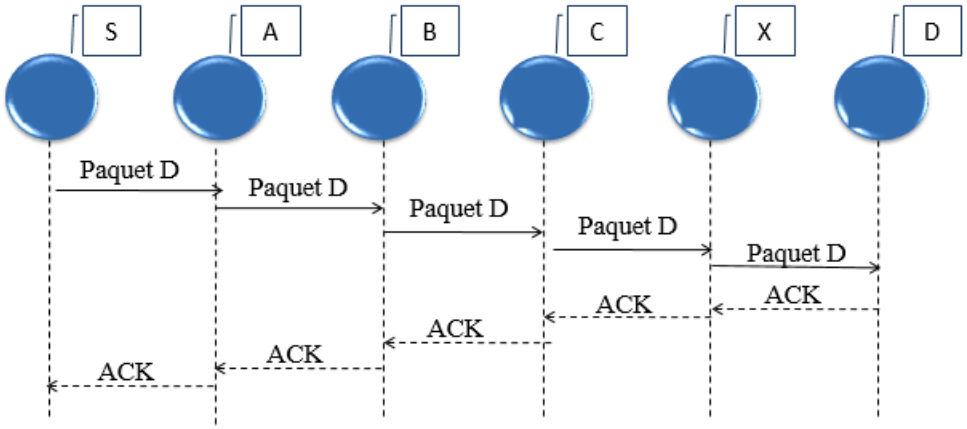


FIGURE 2.5 – Schéma ACK de bout en bout

Le nœud source S envoie le paquet D et tous les nœuds intermédiaires transmettent simplement ce paquet. Lorsque le nœud de destination D reçoit ce paquet D, il renvoie un paquet accusé de réception ACK au nœud source S en utilisant la route inversée des paquets de données. Si le nœud S ne reçoit pas un paquet ACK pour un paquet de donnée, il bascule

vers le mode TACK. Le mode TACK est similaire à l'approche 2ACK. Le mode TACK est utilisé jusqu'à la destination, reçoit un paquet de donnée correctement (ce qui signifie qu'aucune activité malveillante n'est détectée le long de la route). Dans ce cas, la source bascule vers le mode AACK.

EAACK

L'approche EAACK a été proposée dans [32]. L'approche EAACK est le résultat de combinaison de trois modes : AACK, S-ACK et MRA. Les deux modes ACK et S-ACK sont identiques avec les modes AACK et TACK de l'approche [31]. Sauf que dans l'approche EAACK, les auteurs ont incorporé un mécanisme de signature numérique basée sur les algorithmes DSA [33] et RSA [34]. Le mode MRA (Misbehavior Report Authentication) est utilisé pour faire face à l'attaque de fausse accusation. L'utilisation de ce mode nécessite l'utilisation d'une route de transmission disjointe par rapport à la route de transmission. Si un nœud signale un lien malveillant à la source, la destination bascule vers le mode MRA pour vérifier si le rapport de malveillance est valide ou non. La source envoie un paquet MRA à la source en y identifiant le nombre de paquets supprimés. Si la destination a rééllement reçu ces paquets, la source conclut qu'il s'agit d'un faux rapport de malveillance. Autrement, il s'agit d'un rapport valide.

Bien que les approches basées sur les acquittement permettent de résoudre plusieurs limitations des approches basées sur le mode promiscous, elles présentent aussi plusieurs limitations qui peuvent influencer leurs performances. Afin d'illustrer ces limitations, nous prenons la triplette de nœud (A,B,C) comme exemple.

1. Ces approches ne permettent d'identifier que les liens malveillants au lieu des nœuds malveillants (voir la figure 2.6). Cette limitation donne aux nœuds malveillants plus d'opportunités pour supprimer beaucoup de paquets de données en s'impliquant dans plusieurs routes de transmission. Cette limitation peut être exploitée par les nœuds malveillants dans deux comportement différents :

   Comportement 1 : Un nœud malveillant peut lancer l'attaque Black Hole en envoyant une fausse RREP afin d'obliger la source à router les paquets à travers lui. Vu que ces approches ne permettent que détecter le lien malveillant, tous les paquets passant par cette route seront perdus.

   Comportement 2 : Un réseau mobile Ad hoc est un réseau dynamique, ce qui signifie que la topologie du réseau change fréquemment. Un changement de topologie signifie qu'il y a eu un changement dans le voisinage de chaque nœud . Vu que ces approches ne permettent que de détecter les liens malveillants, chaque nouveau voisin

d'un nœud malveillant constitue une nouvelle chance de constituer un lien malveillant, et par conséquent, supprimer plus de paquets de données.

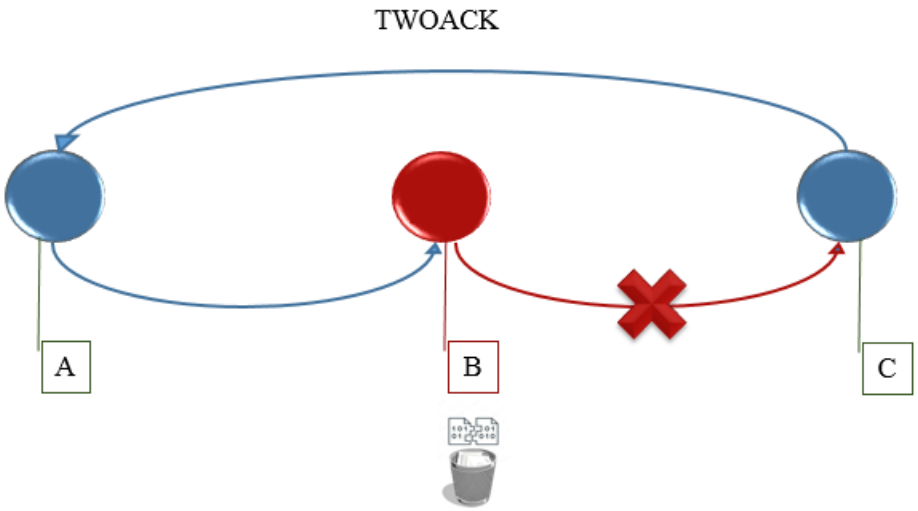


FIGURE 2.6 – Détection des liens malveillants

2. Ces approches ne permettent pas de contrecarrer l'attaque de collusion. Si deux nœuds consécutives dans une route de transmission collaborent pour cacher leurs comportement malveillants, ces approches peuvent être contournées, si les deux nœuds B et C sont en collusion (voir le cas de la figure 2.7).

   Comportement collusion : Le nœud B supprime tous les paquets de données passant par lui. Afin de cacher le comportement malveillant du nœud B, le nœud C renvoie des paquets TWOACK au nœud A pour tous les paquets de données supprimés par le nœud B. Ces approches ne permettent pas de détecter un tel comportement car les paquets de données sont acquittés correctement.

3. Ces approches ne permettent pas de motiver les nœuds à acheminer les paquets. Les nœuds égoïstes peuvent ne pas participer dans le processus de découverte routes afin d'éviter de s'impliquer dans les routes de transmission, et par conséquent, ils évitent de consommer leurs ressources pour les autres. D'un coté, les nœuds égoïstes n'acheminent pas les paquets des autres. De l'autre coté, les nœuds coopératifs routent les paquets initiés par les nœuds égoïstes, ce qui constitue une inégalité entre les nœuds du réseau.

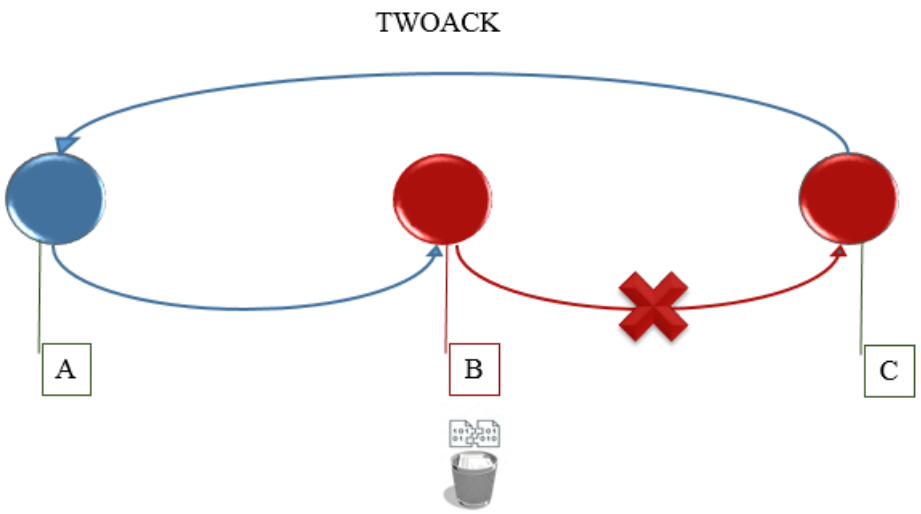


FIGURE 2.7 – Les nœuds B et C sont en collusion

2.6 Conclusion :

Dans ce chapitre, nous avons mis en évidence la problématique de sécurité dans les réseaux mobiles ad hoc. Nous avons présenté l'attaque de suppression de paquets dans le cas du protocole de routage DSR. Nous avons aussi présenté les approches proposées pour contrcarrer à l'attaque de suppression de paquets. Parmi ces approches nous avons mis l'accent sur les approches de réputation basées sur l'acquittement. Ces approches souffrent de plusieurs limitations qui peuvent influencer leurs performance. Dans notre mémoire, nous allons proposer une approche afin de rémédier à ces limitations.

Chapitre 3

Nouvelle approche hybride contre la malveillance et l'égoïsme

3.1 Introduction

Dans ce chapitre, nous présentons notre approche appelée IAACK (Improved AACK). L'approche IAACK est une amélioration de l'approche AACK [31]. En comparaison à l'approche AACK, l'approche IAACK vise à détecter les nœuds malveillants et les punir sévèrement, tout en motivant l'acheminement des paquets par les nœuds égoïstes (inciter les nœuds égoïstes à coopérer). L'approche IAACK est structurée autour de trois composants : surveillance, réputation et stimulation. Le composant de surveillance est résponsable de surveiller l'acheminement des paquets de données par les nœuds voisins afin de détecter des éventuelles suppressions de paquets, ou bien des collusions entre les nœuds. Le composant de réputation est résponsable de quantifier le comportement des nœuds voisins par des valeurs de réputation uniques. Ainsi, chaque nœud ayant une valeur de réputation inférieure au seuil prédéfini est considéré comme malveillant. Le composant de stimulation est responsable de stimuler la coopération des nœuds dans le processus de découverte de route. Pour y faire, nous avons utilisé les crédits comme moyen d'incitation de coopération.

3.2 Notations et modèle du réseau

Nous avons modélisé le réseau mobile ad hoc sous forme d'un graphe orienté $G = (V, E)$, où $V = \{N_1, N_2, ..., N_n\}$ fait référence à l'ensemble des nœuds mobiles du réseau, et E reflète l'ensemble des liens de transmissions. Nous supposons que les paquets d'acquittement échangés entre les nœuds sont signés de manière similaire à l'approche EAACK [32] (existence d'un mécanisme de signature numérique). A noter que notre approche ne nécessite

pas l'incorporation d'une entité centrale de gestion, donc elle fonctionne dans un environnement complètement distribué. Toutes les notations utilisées dans ce chapitre sont résumées au tableau 3.1.

Notations	Description
C_j^i	Nombre de crédis associé au nœud N_i tel qu'il est perçu par le nœud N_j
R_j^i	Récompense en crédits associés au nœud N_i par le nœud N_j
p_j^i	Paiement en crédits déduit du compte du nœud N_i par le nœud N_j
V_j	Ensemble des nœuds voisins directs du nœud N_j
P	Route de transmission où $\{N_s, ...N_i, N_j, N_k....., N_d\}$
$T1$	Minuterie de réception du paquet ACK
$T2$	Minuterie de réception du paquet TACK
D	Paquet de donnée
$Col - numb$	Champs collusion-number
$Col\_dec(N_j, N_k)$	Nombre de collusions détectées à travers le liens de transmission (N_j,N_k)
$Init$	Valeur de réputation initiale
Rep_i^j	Valeur de réputation du nœud N_j tel qu'elle est perçue par le nœud N_i
Dec	Valeur de décrémentation de la réputation d'un nœud
Inc	Valeur d'incrémentation de la réputation d'un nœud
max	Valeur maximale de réputation.
R_{th}	Seuil de réputation prédéfini
Col_{th}	Seuil prédéfini de collusion

TABLE 3.1 – Notations

3.3 Modèle d'attaque

Suivant son but, un nœud peut lancer l'attaque de suppression de paquets soit pour : **(1)** perturber le processus d'acheminement des paquets de données, dans ce cas, on parle

d'un nœud malveillant. **(2)** préserver ses ressources vue que la transmission des paquets lui consomme des ressources. Dans notre cas, les nœuds malveillants participent dans le processus de découverte de routes, mais une fois qu'ils font partie d'une route, ils commencent à supprimer tous les paquets de données destinés à être acheminés. En revanche, les nœuds égoïstes ne participent pas dans le processus de découverte de route. Ils suppriment tous les paquets de routage passant par eux afin d'éviter de s'impliquer dans les routes de transmissions.

3.4 Approche IAACK

Afin de remédier aux limitations des approches basées sur les acquittements décrites précédemment dans le chapitre 2. Nous avons proposé une nouvelle approche dénommée IAACK (Improved AACK) qui est organisée autour de trois composant : Le composant de stimulation, le composant de surveillance et le composant de réputation, qui seront détaillés dans ce qui suit :

3.4.1 Composant de stimulation

Le composant de stimulation est responsable de stimuler les nœuds à coopérer dans le processus de découverte de routes. De manière similaire à l'approche proposée dans [23], nous employons une approche totalement distribuée, ce qui signifie qu'aucune entité centrale n'est impliquée dans le processus de mise à jour des crédits des nœuds du réseau. Dans notre approche, chaque nœud maintient un compteur de crédit pour chaque nœud voisin connu dans le processus de découverte de routes. En comparaison avec les approches existantes, les crédits dans notre approche sont utilisés comme un moyen d'incitation dans le processus de découverte de route et non pas dans le processus de transmission des paquets de données. Un nœud gagne des crédits en acheminant des paquets de routage (RREQ, RREP, RERR). Autrement, un nœud perd des crédits quand il génère des demandes de route (RREQ) comme source. Soit CR_0 le crédit initial attribué à chaque nœud nouveau voisin, et soit CR_j^i reflète le nombre de crédits associé au nœud N_i tel qu'il est perçu par le nœud N_j. Le compteur de crédit d'un nœud est mis à jour suivant son comportement.

Soit RC_j^i la récompense en crédits associés au nœud N_i par le nœud N_j pour chaque paquet de routage acheminé correctement par le nœud N_i. Ainsi, soit PR_j^i le paiement en crédits déduit du compte du nœud N_i par le nœud N_j pour chaque paquet de routage RREQ

généré (comme source) par le nœud N_i et acheminé par le nœud N_j.

Afin d'illustrer le fonctionnement du module de stimulation, nous prenons l'interaction entre une paire de nœuds (N_i, N_j) comme exemple.

Pour chaque paquet de routage $ROUT$ reçu par le nœud N_j depuis le nœud N_i, les comptes de crédits des nœuds N_j et N_i sont mis à jour suivant le processus suivant :

- **Cas 1 :** lorsque N_j reçoit un paquet de routage RREQ comme récepteur ou dans le mode promiscuous, et la source N_i du paquet est un voisin directe du nœud N_j ($N_i \in V_j$ où V_j reflète l'ensemble des nœuds voisins directs du nœud N_j). Dans ce cas, N_j achemine le paquet de routage RREQ et extrait des crédits du compte de crédit du nœud N_i suivant la formule suivante :

$$CR_j^i = CR_j^i - PR_j^i(ROUT) \tag{3.1}$$

- **Cas 2 :** lorsque N_j reçoit un paquet de routage $ROUT$ de type RREQ, RREP, RERR comme récepteur ou dans le mode promiscuous initié par le nœud N_k et acheminé par le nœud voisin N_i ($N_i \in V_j$), le nœud N_j récompense le nœud N_i pour l'acheminement correct du paquet R suivant la formule suivante :

$$CR_j^i = CR_j^i + RC_j^i(ROUT) \tag{3.2}$$

3.4.1.1 Isolation des nœuds égoïstes

Chaque nœud ayant un compteur de crédit égal à zéro est considéré comme un nœud égoïste. Afin de les punir, les nœuds honnêtes refusent d'acheminer les paquets RREQ générés par les nœuds égoïstes jusqu'à ce qu'ils deviennent plus coopératifs. En routant les paquets de routage correctement, les nœuds égoïstes peuvent améliorer leurs compteurs de crédits, ce qui leur permet de faire router leurs paquets RREQ.

Discussion

Notre approche utilise les crédits comme moyen d'incitation afin de motiver les nœuds à coopérer dans le processus de découverte de route. Les nœuds égoïstes refusant de coopérer sont stimulés à router les paquets de routage, car les nœuds refusant de les router ne peuvent

pas gagner des crédits. Par conséquent, ils ne peuvent pas acheminer leurs paquets RREQ, ce qui veut dire qu' ils ne peuvent pas faire acheminer leurs paquets de données. Donc, afin que les nœuds puissent faire acheminer leurs paquets, ils doivent coopérer en routant des paquets de routage.

3.4.2 Composant de surveillance

Ce composant a comme responsabilité de surveiller le comportement des nœuds voisins dans le processus de transmission des paquets de données, afin de détecter si un nœud est malveillant ou bien coopératif (voir l'algorithme 1). Ce dernier améliore l'approche de surveillance utilisée dans l'approche AACK [31] et l'utilise comme technique de surveillance. Cette approche est le résultat de la combinaison de deux modes : ACK et TACK. Le mode ACK est équivalent au mode d'acquittement de bout-en-bout. Dans ce mode, la destination doit renvoyer un paquet ACK à la source pour chaque paquet de données reçu correctement. Le mode par défaut utilisé est le mode ACK. Cependant, si la source ne reçoit pas un paquet ACK pour un paquet de données avant que le temps d'attente ne s'écoule, la source bascule vers le mode TACK. Le mode TACK est similaire à l'approche TWOACK [30]. Afin d'illustrer le fonctionnement de cette approche de surveillance (voir figure 3.1), soit la triplette de nœuds $< N_i, N_j, N_k >$ $\in$ p avec p $=\{N_s, ... N_i, N_j, N_k, ...N_d\}$ est la route de transmission, où N_s et N_d sont les nœuds source et destination, respectivement.

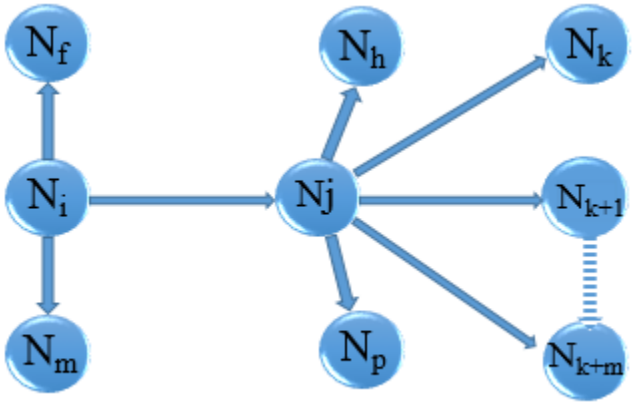


FIGURE 3.1 – Exemple de scénario de surveillance

Lorsqu'il y a des paquets de données à échanger entre les nœuds N_s et N_d, le mode ACK est utilisé par défaut (voir figure 3.2). Si N_s reçoit un paquet ACK pour un paquet de données envoyées avant que le délai $T1$ expire, l'approche de surveillance continue avec le

mode ACK. Autrement, si N_i n'a pas reçu un ACK après que $T1$ ait expiré, N_s bascule vers le mode TACK (voir figure 3.3). Afin de contrecarrer l'attaque collusion, nous avons ajouté un champ signé (signature numérique) dénoté **Collusion-number** (nombre aléatoire généré par le premier nœud de la triplette) à l'entête du paquet de données de manière similaire à l'approche AACK [31]. Ainsi, nous avons ajouté ce même champ à l'entête du paquet TACK. Dans le mode TACK, le troisième nœud de la triplette N_k renvoie un paquet TACK au premier nœud de la triplette N_i pour chaque paquet de donnée reçu.

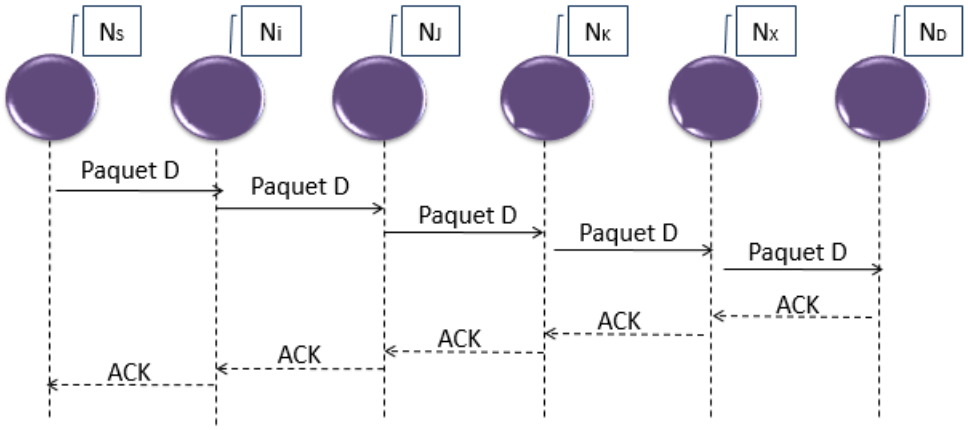


FIGURE 3.2 – Mode ACK

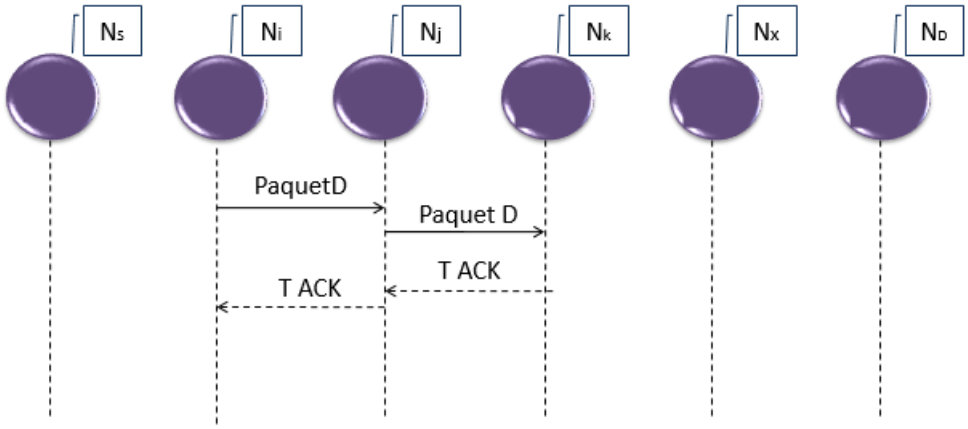


FIGURE 3.3 – Mode TACK

Une fois que le nœud N_k a reçu un paquet de données, il extrait le champ **Collusion-**

number du paquet de données, et il l'insère ensuite dans l'entête du paquet TACK. Puis, il envoie le paquet TACK au nœud N_i. Un événement positif est enregistré contre les deux nœuds N_j et N_k pour le paquet de donnée D, si seulement si les deux conditions suivantes sont vérifiées :

- N_i a reçu un paquet TACK du nœud N_k pour le paquet de données D avant que T2 expire.
- Le champ **Collusion-number** inclus dans l'entête du paquet TACK est égal à celui généré par le nœud N_i.

Tandis que si le paquet de donnée D n'est pas acquitté ($T2$ a expiré) dans les délais par le nœud N_k, le nœud N_i enregistre un événement négatif contre les deux nœudsN_j etN_k . Ainsi, un événement collusion est enregistré dans le cas où le paquet de donnée D est acquitté correctement par le nœud N_k, mais le champ **Collusion-number** inclus dans l'entête du paquet TACK est différent de celui généré par le nœud N_i. Dans ce qui suit, nous présentons un algorithme qui résume le comportement de ce composant.

Algorithme 1 : Algorithme de surveillance

Notations

1. P = $\{N_s, ...N_i, N_j, N_k....., N_d\}$ est la route de transmission ;
2. $T1$: Minuterie de réception du paquet ACK ;
3. $T2$: Minuterie de réception du paquet TACK ;
4. D : Paquet de donnée ;
5. Col_numb : Champs collusion-number contenu dans le paquet $TACK_D$;
6. $Col\_numb(N_i)$: Collusion-number généré par N_i ;
7. **(1) Enregistrement d'un événement positif** ;
8. **Si** N_i reçoit un paquet $TACK_D$ par le nœud N_k **ET** $T2$ n'a pas expiré **ET** $Col\_numb = Col\_numb(N_i)$ **Alors**
9. N_i enregistre un événement **positif** contre les deux nœuds N_j et N_k
10. **FinSi**
11. **(2) Enregistrement d'un événement négatif**
12. **Si** N_i n'a pas reçu un paquet $TACK_D$ par le nœud après que $T2$ a expirée **Alors**
13. N_i enregistre un événement **négatif** contre les deux nœuds N_j et N_k
14. **FinSi**

15. **(3) Enregistrement d'un événement collusion**
16. **Si** N_i reçoit un paquet $TACK_D$ par le nœud N_k **ET** $T2$ n'a pas expirée **ET** $Col\_numb \neq Col\_numb(N_i)$ **Alors**
17. N_i enregistre un événement **collusion** contre les deux nœuds N_j et N_k
18. **FinSi**

3.4.3 Composant de Réputation

Le composant de réputation est responsable de l'évaluation et de la mise à jour des valeurs de réputation des nœuds voisins dans le processus de transmission des paquets de données. Il a pour mission d'isoler les nœuds malveillants détectés. Ce dernier est composé de deux processus complémentaires : le processus de calcul de réputation et le processus d'isolation.

3.4.3.1 Le processus de calcul de réputation

Le processus de calcul de réputation permet de quantifier le comportement des voisins dans le processus de transmission des paquets de données par une valeur de réputation unique. En comparaison avec les approches d'acquittement existantes, on évalue la réputation des nœuds et non pas la réputation des liens. Dans notre approche, la réputation d'un nœud reflète sa fiabilité dans tous les liens de communications dans lequel il est impliqué.

Afin d'illustrer le fonctionnement du composant de réputation, nous prenons la triplette de nœud $< n_i, n_j, n_k > \in P$ comme exemple, où P est une route de transmission. Soit Rep_i^j et Rep_i^k les valeurs de réputation des nœuds n_j et n_k tel qu'elles sont perçues par le nœud N_i. Ainsi, le nœud N_i maintient un compteur de collusion dénoté **Col-dec**(n_j, n_k) qui reflète le nombre de collusions détectés à travers le liens de transmission (n_j, n_k).

Au démarrage, la valeur de réputation de chaque nœud surveillé est initialisée à la valeur *init* et elle varie entre 0 et *max* ($max \geq 1$). Suivant le type d'événement détecté à travers le lien (n_j, n_k) par le composant de surveillance, les valeurs de réputations Rep_i^j et Rep_i^k, et le compteur **Col-dec**(n_j, n_k) sont mis à jour.

Événement positif

Si le composant de surveillance du nœud N_i a détecté un événement positif à travers le lien (n_j, n_k) , les valeurs de réputation Rep_i^j et Rep_i^k des nœuds n_j et n_k sont incrémentées par la valeur Inc (équations (3.3) et (3.4)) par le composant de réputation :

$$Rep_i^j = Rep_i^j + Inc \tag{3.3}$$

$$Rep_i^k = Rep_i^k + Inc \tag{3.4}$$

Événement négatif

Si le composant de surveillance du nœud N_i a détecté un événement négatif à travers le lien (N_j, N_k), les valeurs de réputation Rep_i^j et Rep_i^k des nœuds N_j et N_k sont décrémentées par DEC_i^j et DEC_i^k par le composant de réputation :

$$Rep_i^j = Rep_i^j - DEC_i^j \tag{3.5}$$

$$Rep_i^k = Rep_i^k - DEC_i^k \tag{3.6}$$

Dans notre approche, nous faisons une distinction entre la valeur de l'incrémentation et la valeur de décrémentation de la réputation. Le but de cette idée est de ne pas traiter de la même manière les nœuds ayant des valeurs de réputation élevées et les nœuds ayant des valeurs de réputation faibles lorsqu'ils font partie d'un même événement négatif. A cet effet, en se basant sur leurs valeurs de réputation, les nœuds sont classés en trois catégories de coopération : coopération élevée, coopération moyenne, coopération faible. Suivant notre approche, la valeur de décrémentation de réputation DEC associé à un nœud dépend de sa valeur de réputation. Soient a, b et c trois constantes utilisées comme valeur de décrémentation de réputation avec a <b <c. Les limites en termes de réputation et la valeur de décrémentation de chaque catégorie de coopération sont présentés dans le tableau suivant (avec rth<init<sup<max).

Réputation	Catégorie de coopération	Valeur de décrémentation DEC
[Sup, max]	coopération élevée	a
[init, Sup[	coopération moyenne	b
]Rth, init[	coopération faible	c

TABLE 3.2 – Valeur de décrémentation

La raison d'être de cette idée est que : Afin d'arriver à leur but qui consiste à déstabiliser le processus de transmission des paquets de données, les nœuds malveillants doivent s'impliquer dans le maximum de routes de transmissions afin de supprimer beaucoup de paquets de données. Ce comportement cause une dégradation dans leurs valeurs de réputation car ils sont impliqués dans beaucoup d'événements négatifs (paquets de données supprimés). Cependant, les nœuds coopératifs sont caractérisés par leurs valeurs de réputation élevées car ils collectent beaucoup d'événements positifs dus à la transmission correcte des paquets de données destinés à être acheminés. Dans notre approche, lorsqu'un nœud coopératif (réputation élevée) et un nœud ayant une réputation faible (probablement malveillant) sont impliqués dans un même événement négatif, ils ne sont pas traités de la même manière, mais ils sont traités suivant leurs valeurs de réputation. La valeur de réputation d'un nœud coopératif est décrémentée par une faible valeur DEC, tandis que la valeur de réputation d'un nœud faible-réputé est décrémenté par une valeur élevé DEC, ce qui cause la dégradation de sa valeur de réputation. Avec cette méthode, nous avons assuré une équité (égalité) entre un nœud coopératif et un nœud malveillant lorsqu'ils sont impliqués dans le même événement négatif.

Si la valeur de réputation d'un nœud est supérieure à un certain seuil prédéfini R_{th}, le processus d'isolation est invoqué.

Evènement Collusion

Lorsque le composant de surveillance du nœud N_i a détecté un événement Collusion à travers le lien (N_j,N_k), le composant de réputation incrémente le compteur **Col-dec(**N_j, N_k**)** associé au lien (N_j,N_k). A noter que le compteur **Col-dec(**N_j**,**N_k**)** est initialisé au début à 0. Si **Col-dec(**N_j**,**N_k**)** est supérieur à un seuil prédéfini Col_{th}, le processus d'isolation est invoqué.

3.4.3.2 Processus d'isolation

Le processus d'isolation a pour mission d'isoler les nœuds malveillants et les nœuds en collusion après leurs identifications.

Isolation des nœuds malveillants

Si la réputation d'un nœud Rep_i^j est inférieure au seuil prédéfini R_{th}, le nœud N_j est considéré comme malveillant. Pour son isolation, le nœud N_i procède aux actions suivantes :

1. Envoie un rapport de malveillance à la source pour l'informer du nœud malveillant détecté.

2. Ajoute le nœud malveillant détecté à sa liste noire des nœuds malveillants isolés.
3. Invalide toutes les routes de transmission impliquant le nœud détecté.
4. Refuse de router toutes les RREQ initiées par ce nœud pour sa punition.

Chaque nœud y compris la source recevant le rapport de malveillance dans le mode promiscuous ou comme récepteur procède au même processus d'isolation décrit dans les actions précédentes.

Isolation des nœuds en collusion

Si le compteur **Col-dec** (N_j, N_k) est supérieur à un seuil prédéfini Col_{th}, les deux nœuds N_j et N_k sont considérés comme des nœuds en collusion par le nœud N_i. Pour les isoler, le nœud N_i informe la source en envoyant un rapport de collusion en y identifiant les deux nœuds détectés. Le nœud N_i ajoute les deux nœuds détectés à la liste noire des collusions et invalide toute les routes impliquant ces deux nœuds. Pour les punir, les nœuds coopératifs refusent de router les RREQ initiés par les nœuds présents dans la liste noire des collusions. Ainsi, de manière similaire au processus précédent, chaque nœud y compris la source recevant le rapport de collusion dans le mode promiscuous ou comme récepteur procède à la même isolation du nœud N_i.

3.5 Conclusion

Dans ce chapitre, nous avons présenté notre approche IAACK, qui est une approche qui vise à punir les nœuds malveillants, tout en motivant la coopération des nœuds égoïstes. Ce but est atteint par l'incorporation des composants de stimulation, surveillance et réputation. Nous avons proposé une technique qui nous permet d'identifier les nœuds malveillants au lieu des liens malveillants. Ainsi, nous avons modifié l'approche de surveillance AACK afin de la rendre résistante à l'attaque collusion. Notre approche emploie des crédits comme incitation pour motiver les nœuds égoïstes à coopérer dans le processus de découverte de route. Dans le chapitre suivant, nous présenterons l'évaluation de performance de notre approche en utilisant le simulateur NS 2.34 [35].

Chapitre 4

Simulation et évaluation de performance

4.1 Introduction

Après avoir détaillé le fonctionnement de notre approche IAACK, nous présentons dans ce qui suit les étapes de simulation ainsi qu'une interprétation détaillée des résultats de simulation sous forme d'une étude comparative entre notre approche IAACK et l'approche AACK [31].

4.2 Aperçu sur les simulateurs de réseaux Informatique

Parmi la multitude de simulateurs de réseau existants, nous avons retenu quatre d'entre eux qui seront présentés ci dessous.

GloMoSim (Global Mobile Information System Simulator)

GloMoSim [36] est un simulateur conçu à l'origine pour la simulation des réseaux mobiles à grande échelle. Il est construit à partir du langage Parsec (Parallel Simulation Environment for Complex System). Ce dernier est un langage de simulation parallèle dérivé du language de programation C auquel sont rajouté des fonctions d'envoi, de réception de message et de gestion de timer. GloMoSim constitue l'un des simulateurs les plus riches. Il se vend en version commerciale sous le nom de QualNet [37].

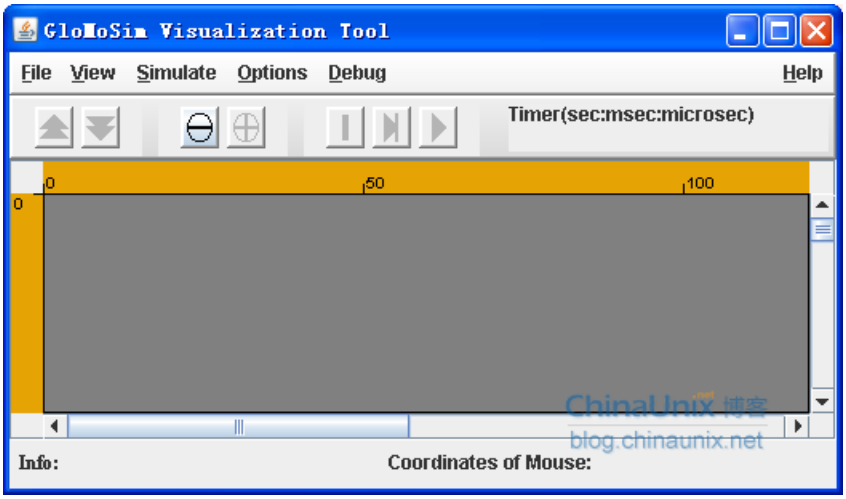


FIGURE 4.1 – GloMoSim (Global Mobile Information System Simulator)

Omnet++ (Objective Modular Network)

OMNET++ [38], est un projet open source dont le développement à commencé en 1992 par Andras Vargas à l'Universités de Budapest. Actuellement, ce simulateur est utilisé par des dizaines d'universités pour la validation de nouveaux matériels et logiciels, ainsi que pour l'analyse de performance et l'évaluation de protocoles de communication. L'avantage de OMNET ++ est sa facilité d'apprentissage, d'intégration de nouveaux modules et la modification de ceux déjà implémentés.

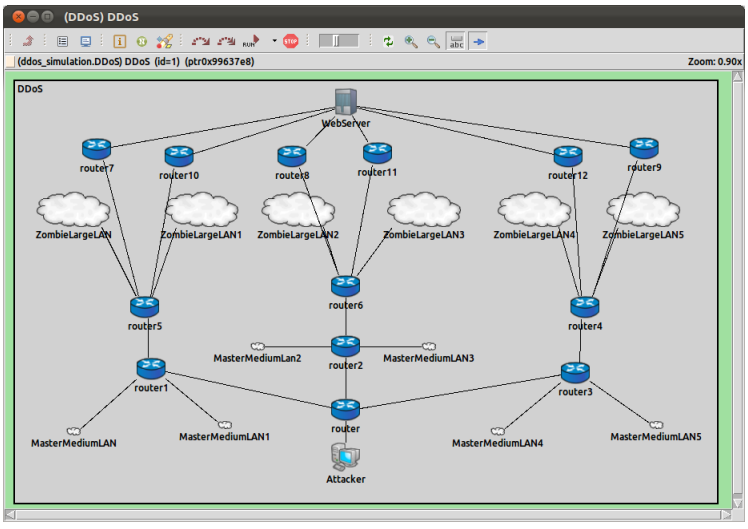


FIGURE 4.2 – Omnet++ (Objective Modular Network)

J-Sim (JavaSim)

J-Sim[39], autrefois connu sous le nom de JavaSim, est un simulateur développé en Java. Il offre une bibliothèque de simulation de réseaux de capteurs. Son architecture se subdivise en trois types de composants : un premier pour contenir la représentation des nœuds, un deuxième pour la représentation du canal de captage et un troisième pour la représentation du canal radio.

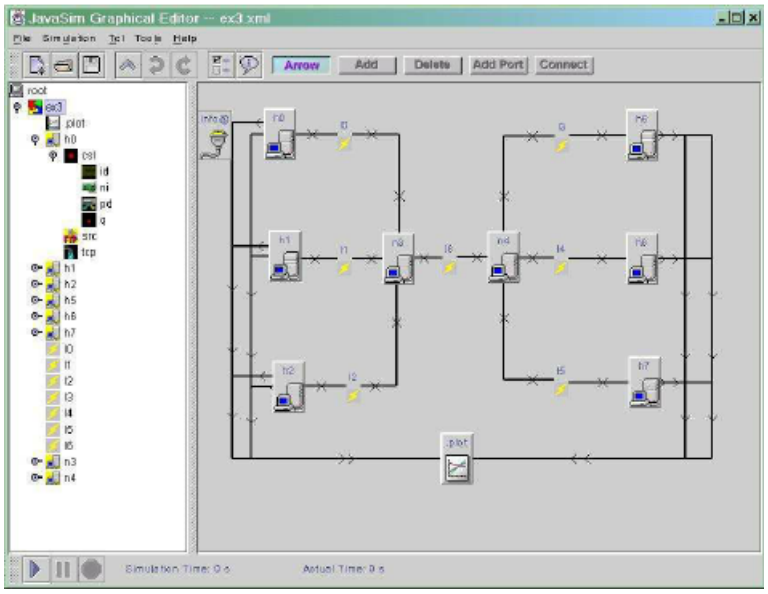


FIGURE 4.3 – J-Sim (JavaSim)

NS-2 (Network Simulator 2)

NS-2 [35]est certainement le plus populaire des simulateurs de réseaux. Il est conçu à partir du langage C++. Sa richesse réside dans sa ré-utilisabilité et sa modularité. De nombreux modèles déjà disponibles, parmi lesquels des modules destinés à la simulation de réseaux sans fil. Nous pouvons aussi remarquer que le contrôle de la simulation se fait à l'aide d'un langage de script Object Tool Command Language (OTCL). L'outil NAM (Network Animator) permet de visualiser des animations de la simulation (transfert des paquets d'un nœud à un autre, taille des paquets, etc...).

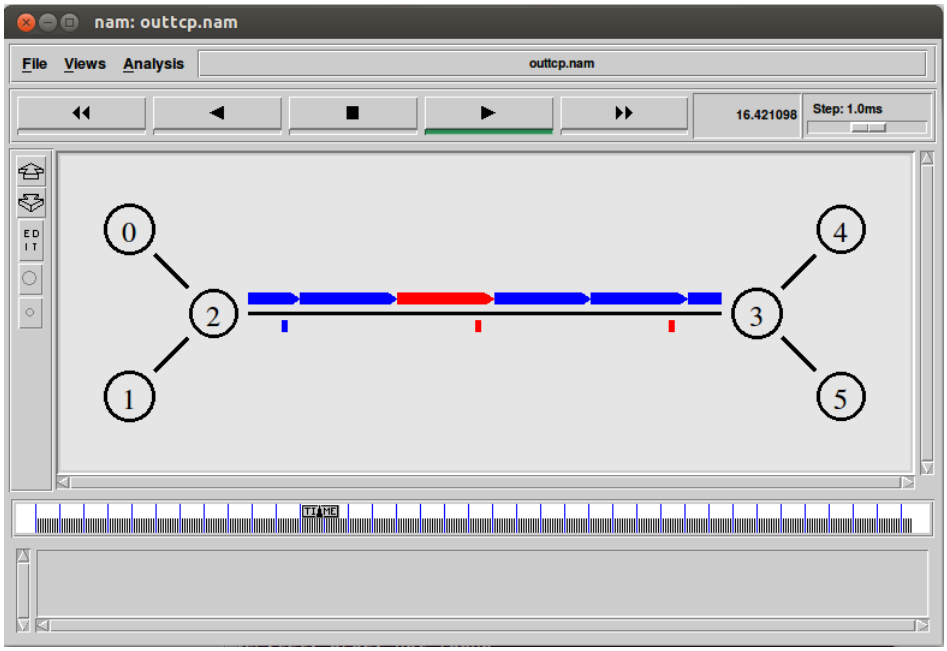


FIGURE 4.4 – NS-2 (Network Simulator 2)

4.2.1 Choix de NS 2

Après avoir décrit les plus importants simulateurs dans la section précédente, Il en résulte que NS2 reste sans doute le simulateur le plus documenté et le plus polyvalent [40]. Nous l'avons retenu afin de simuler la nouvelle approche que nous avons proposé.

4.3 Présentation de NS2

Le Network Simulator 2 est un ensemble d'outils qui simule le comportement du réseau. Il permet de créer des topologies réseaux et aussi d'analyser ces événements afin de comprendre son comportement . Le simulateur utilise le langage orienté objet OTCL dérivé de TCL pour la description des conditions de simulation sous forme de script en fournissant les caractéristiques des liens physiques, les protocoles utilisés, le type de trafic généré par les sources, etc. L'observation de ce comportement se fait via l'outil NAM. Pour cela nous avons décidé d'approfondir un peu plus son utilisation. Dans un premier temps, nous présenterons le langage de script tcl, l'outil de simulation NS2 et l'outil de virtualisation NAM.Pour finir nous allons décrire les étapes à suivre afin d'insérer un protocole de routage dans NS.

4.3.1 Le langage de script TCL

Le langage TCL est un langage de script puissant qui permet d'utiliser éventuellement une approche de programmation orientée objet. Il est facilement extensible par un certain nombre de modules. Dans notre cas, il est indispensable d'utiliser le langage TCL pour pouvoir travailler avec les objets fournis par NS-2.
L'OTCL est un langage dérivé du langage TCL avec les extensions orientée objet. NS2 utilise otcl pour les simulation afin de créer les objets de réseau dans la mémoire et d'insérer des événements dans la file d'attente.

4.3.2 L'outil de simulation NS-2

L'outil NS-2 fournit un ensemble d'objets TCL spécialement adaptés à la simulation de réseaux. Avec les objets proposés par ce moteur, on peut représenter des réseaux avec liens filaires ou sans fils, des machines et routeurs (Node), des flux TCP et UDP (par exemple pour simuler un flux CBR1), et définir les règles régissant les files d'attentes mises en œuvre dans chacun des nœuds. NS-2 ne permet pas de visualiser le résultat des expérimentations. Il permet uniquement de stocker une trace de la simulation, de sorte qu'elle puisse être exploitée par un autre logiciel, comme le NAM.

4.3.3 L'outil de visualisation NAM

NAM est un outil de visualisation qui présente deux intérêts principaux : représenter la topologie d'un réseau décrite avec NS-2, et afficher temporellement les résultats d'une trace d'exécution NS-2. Par exemple, il est capable de représenter des paquets TCP ou UDP, la rupture d'un lien entre nœuds, ou encore de représenter les paquets rejetés d'une file d'attente pleine. Ce logiciel est souvent appelé directement depuis les scripts TCL pour NS-2, de sorte à visualiser directement le résultat de la simulation.

4.4 Étape d'implémentation de l'approche IAACK

Dans cette section, nous allons décrire la procédure d'implémentation de notre approche IAACK sous NS2.34 .
Etape 1 : Nous avons créé un répertoire de protocole de routage nommé IAACK-DSR. Dans le répertoire, on a créé trois fichiers IAACK-DSR.cc ,IAACK-DSR.h,IAACK-DSR_ packt.h.
Etape 2 : Nous avons extrait les fichiers suivants afin de les modifier :

1. **$NS_ ROOT / Makefile**

2. **$NS_ ROOT / queue / priqueue.cc**
3. **$NS_ ROOT / common / packet.h**
4. **$NS_ ROOT / trace / cmu-trace.h**
5. **$NS_ ROOT / trace / cmu-trace.cc**
6. **$NS_ ROOT / tcl / lib / ns-packet.tcl**
7. **$NS_ ROOT / tcl / lib / ns-lib.tcl**
8. **$NS_ ROOT / tcl / lib / ns-agent.tcl**
9. **$NS_ ROOT / tcl / lib / ns-mobilenode.tcl**

Dans ce qui suit, nous allons expliquer les différentes modifications apportées à ces fichiers. Nous commencerons par le premier fichier ˜/ ns-allinone-2.34 / ns-2.34 / Makefile auquel nous avons rajouté le privilège suivant à la ligne 269 :

- IAACK-DSR/IAACK-DSR.o

Etape 3 : nous avons rajouté en suite la ligne suivante à ˜/ns-allinone-2.34/ ns-2.34/queue/priqueue.cc à partir de la ligne 93/

- PT IAACK-DSR

Etape 4 : Pour définir un nouveau type de paquet de protocole de routage, nous avons dû modifier le fichier˜/ ns-allinone-2.34 / ns-2.34 / common / packet.h . en modifiant PT_ NTYPE en 63 et pour notre protocole PT_ IAACK-DSR en 62. À partir de la ligne 85 les corrections seront comme suit :

1. //IAACK-DSR packet
2. static const packet _t PT _IAACK-DSR = 62 ;
3. //insert new packet types here
4. static packet_ t PT_ NTYPE = 63 ;

Etape 5 : Nous avons effectué le changement du fichier ˜/ ns-allinone-2.34 / ns-2.34 / common / packet.h. Le changement est effectué à la ligne 254 :

1. type = PT_ AODV
2. type = PT_ IAACK-DSR

Etape 6 : Et à la ligne 390 du même fichier les modifications sont les suivantes :

1. //IAACK-DSR patch
2. name_[PT_ IAACK-DSR] = "IAACK-DSR" ;

Etape 7 : Afin de faire en sorte que NS2 trace notre simulation et puisse l'écrire dans le fichier tracetr , nous avons dû modifier les deux fichier cmu-trace.h et cmu-trace.cc

Etape 8 : Pour intégrer une fonction de trace, nous avons ajouté la ligne suivante à ~/ ns-allinone-2.34/ns-2.34/trace/cmu-trace.h à la ligne 163 :

- void format_ IAACK-DSR(Packet *p, int offset) ;

Etape 9 : Le fichier~/Ns-allinone-2.34/ns-2.34/trace/cmu-trace.cc à été ajouté à la ligne 1071 par les instructions suivantes :

1. // IAACK-DSR patch
2. void
3. CMUTrace : :format_ IAACK-DSR(Packet *p, int offset) ;
4. {
5. struct hdr_ IAACK-DSR*wh = HDR_ IAACK-DSR(p) ;
6. struct hdr_ IAACK-DSR_ beacon *wb = HDR_ IAACK-DSR_ BEACON(p) ;
7. struct hdr_ IAACK-DSR_ error *we = HDR_IAACK-DSR_ ERROR(p) ;
8. switch(wh- pkt_ type) {
9. case IAACK-DSR_ BEACON :
10. if (pt_ ⇒tagged()) {
11. sprintf(pt _ ⇒buffer() + offset,"-IAACK-DSR :t % x IAACK-DSR :h % d IAACK-DSR :b % d IAACK-DSR :s %d "
12. "-IAACK-DSR :px %d - IAACK-DSR :py %d _IAACK-DSR :ts %f " ;
13. "-IAACK-DSR :c BEACON ", ;

```
14. wb⇒pkt_ type ;
15. wb⇒beacon_ hops ;
16. wb⇒beacon_ id ;
17. wb⇒beacon_ src ;
18. wb⇒beacon_ posx ;
19. wb⇒beacon_ posy ;
20. wb⇒timestamp) ;
21. } else if (newtrace_) {
22. printf(pt_ ⇒buffer() + offset ;
23. "-P IAACK-DSR -Pt 0x%x -Ph %d -Pb %d -Ps %d -Ppx %d -Ppy %d -Pts %f Pc
    BEACON " ;
24. wb⇒pkt_ type ;
25. wb⇒beacon_ hops ;
26. wb⇒beacon_ id ;
27. wb⇒beacon_ src ;
28. wb⇒beacon_ posx ;
29. wb⇒beacon_ posy ;
30. wb⇒timestamp) ;
31. } else {
32. sprintf(pt_ ⇒buffer() + offset, "[0x%x %d %d [%d %d] [%d %f]] (BEACON)" ;
33. wb⇒pkt_ type ;
34. wb⇒beacon_ hops ;
35. wb⇒ beacon_ id ;
36. wb⇒beacon_ src ;
37. wb⇒beacon_ posx ;
38. wb⇒beacon_ posy ;
39. wb⇒timestamp) ;
40. }
41. break ;
```

42. case IAACK-DSR_ ERROR :
43. break ;
44. default :
45. # ifdef WIN32
46. fprintf(stderr,CMUTrace : :format_ IAACK-DSR : invalid IAACK-DSR packet typen") ;
47. # else
48. printf(stderr,"%s : invalid IAACK-DSR packet typen", _ _ FUNCTION_ _) ;
49. # end if
50. abort() ;
51. }

Etape 10 : Nous avons modifié les fichiers tcl de manière à créer un agent de routage en définissant d'abord le nom du protocole à utiliser dans le fichier tcl. Cela s'est fait en modifiant / ns-allinone-2.34 / ns-2.34 / tcl / lib / ns-packet.tcl a la ligne 172.

- IAACK-DSR ;

Etape 11 : Nous avons défini un agent de routage en modifiant / ns-allinone-2.34 / ns-2.34 / tcl / lib / ns-lib.tcl a la ligne 633 par :

1. IAACK-DSR {
2. set ragent [$self create-IAACK-DSR-agent $node]
3. }

Etape 12 : De la ligne 860 du même code de fichier, nous avons ajouté les lignes suivantes :

1. Simulator instproc create-IAACK-DSR-agent { node } {
2. Create IAACK-DSR routing agent ;
3. Set ragent [new Agent/IAACK-DSR [$node node-addr]] ;
4. $self at 0.0 "$ragent start" ;
5. $node set ragent_ $ragent ;
6. return $ragent ;

7. }

Etape 13 : Nous avons défini les numéros de port de l'agent de routage. Le sport est un port source, dport est un port de destination. Pour cela nous avons dû modifier le fichier / ns-allinone-2.34 / ns-2.34 / tcl / lib / ns-agent.tcl a la ligne 202 par :

1. Agent/IAACK-DSR instproc init args { ;
2. $self next $args ;
3. } ;
4. Agent/IAACK-DSR set sport_ 0 ;
5. Agent/IAACK-DSR set dport_ 0 ;

Etape 14 : Pour le dernier fichier / ns-allinone-2.34 / ns-2.34 / tcl / lib / ns-mobilenode.tcl nous avons modifié la ligne 201 :

1. # Special processing for IAACK-DSR
2. set IAACK-DSRonly [string first "IAACK-DSR" [$agent info class]] ;
3. if {$IAACK-DSRonly != -1 } {
4. $agent if-queue [$self set ifq_ (0)] ;}

Etape 15 : Pour la compilation nous avons écrit la commande suivante dans le répertoire / ns-allinone-2.34 / ns-2.34 / :

1. make clean ;
2. make ;

4.5 Simulation et évaluation de performance

Dans cette section, en utilisant le simulateur NS-2.34 [35], nous étudions la performance de notre approche IAACK en comparaison à l'approche AACK [31] dans deux environnements différents : malveillant et égoïste.

4.5.1 Environnements de simulation

Dans l'environnement de simulation malveillant, nous avons évalué la performance de notre approche pour ce qui concerne la détection et l'isolation des nœuds malveillants. Nous avons simulé 40 nœuds déployés de manière aléatoire sur une surface de 670 m × 670 m. Comme source de traffic, le traffic UDP avec CBR (constant bit rate) est utilisé. La norme IEEE 802.11 MAC est utilisé. La portée de transmission de chaque nœud est fixé à 250 m. Le temps de simulation est fixé à 600 s. La valeur initiale de réputation attribée à un nœud au démarage *init* est fixée à 40 et la valeur de réputation de chaque nœud varie entre 1 et 80. Le reste des paramètres de simulations sont présentés dans le tableau 4.1.

Paramètre	Valeur
Nombre de nœuds	40
Protocole de routage	DSR
Surface de simulation	670 m * 670 m
Portée de transmission	250 m
Vitesse des nœuds	10m/s et 20m/s
Temps de pause	0 s
Nombre de nœuds malveillants	2 ; 4 ; 6 ; 8 ; 10 ; 12
Modèle de mobilité	Random Way Point
Nombre de CBR	10 connections
Temps de simulation	600 s

TABLE 4.1 – Paramètres de simulation de l'environnement malveillant

Dans l'environnement de simulation égoïste, nous avons examiné la performance de notre approche pour ce qui concerne l'isolation des nœuds égoïstes. Nous avons simulé 40 nœuds déployés de manière aléatoire sur une surface de 1000 m × 1000 m. Comme source de trafic, le trafic UDP avec CBR (constant bit rate) est utilisé. La portée de transmission de chaque nœud est fixé à 250 m. Le temps de simulation est fixé à 1200 s. Parmi les 40 nœuds simulés, nous avons choisi 3 nœuds à agir de manière égoïste. Les nœuds égoïstes simulés refusent de participer au processus de découverte de route (supprimer tous les paquets de routage), tout

en essayant de faire acheminer leurs propres paquets. Le reste des paramètres de simulations sont présentés dans le tableau 4.2.

Paramètre	Valeur
Nombre de nœuds	40
Protocole de routage	DSR
Surface de simulation	1000 m * 1000 m
Portée de transmission	250 m
Vitesse des nœuds	2m/s et 20m/s
Temps de pause	0 s
ID des nœuds égoïstes	6,7,8
Modèle de mobilité	Random Way Point
Nombre de CBR	16
Temps de simulation	1200 s

TABLE 4.2 – Paramètres de simulation de l'environnement égoïste

Nous avons utilisé les trois métriques [26] suivantes afin d'examiner la performance de l'approche IAACK :

Débit (Kbps) : La taille totale de tous les paquets de données reçus correctement par les nœuds de destination.

Taux de suppression malveillant : représente le rapport entre le nombre de paquets de données supprimés par les nœuds malveillants et le nombre de paquets de données envoyés.

Taux de succès égotiste : représente le taux de paquets de données acheminés par les nœuds coopératifs pour les nœuds égotistes.

4.5.2 Résultats de simulation

La figure 4.5 trace le débit des approches IAACK et AACK en fonction de la variation du nombre de nœuds malveillants. Dans ce cas de figure, la vitesse des nœuds est fixée à 10 m/s. Nous observons que l'augmentation du nombre de nœuds maleveillants engendre la détérioration du débit des deux approches. Cependant, le débit de l'approche IAACK est supérieur au débit de l'approche AACK. Ceci est dû au fait que l'approche IAACK est

capable de détecter et d'isoler les nœuds malveillants dans le processus de transmission de données au lieu des liens malveillants en comparaison à l'approche AACK.

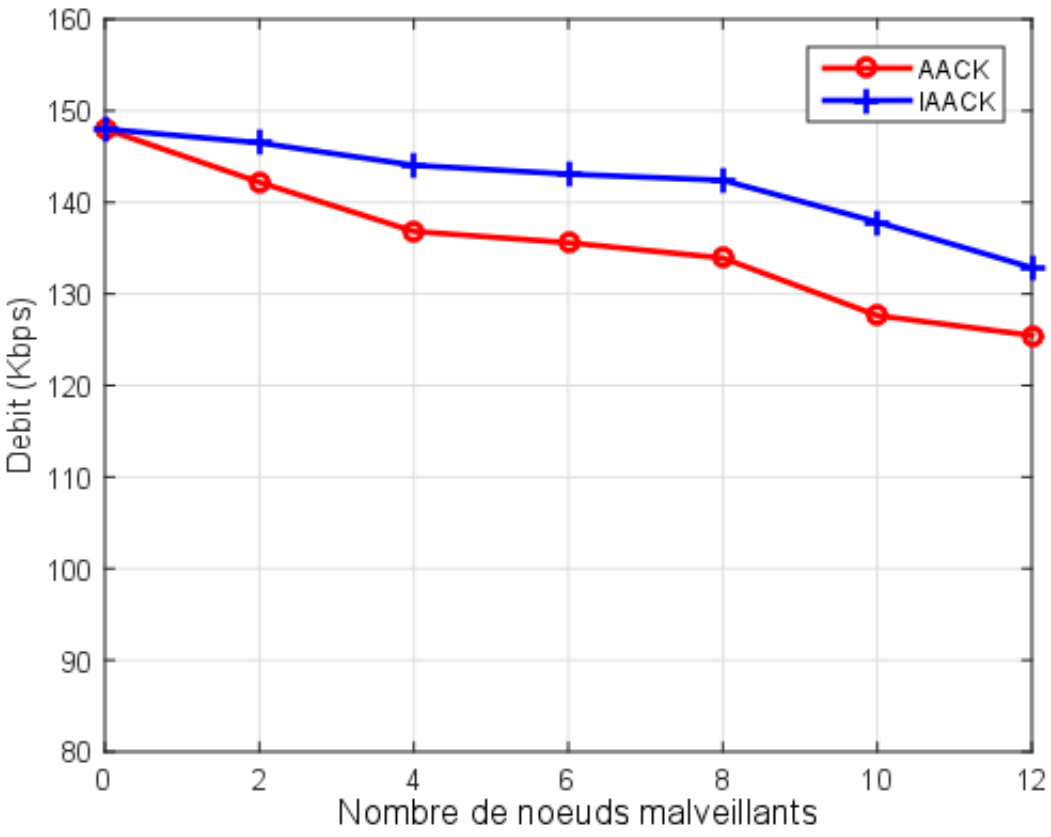


FIGURE 4.5 – Débit Vs nombre de nœuds malveillants (Vitesse = 10 m/s)

La figure 4.6 présente le taux de suppression avec les approches IAACK et AACK en fonction de la variation du nombre de nœuds malveillants. Nous observons que le taux de suppression augmente à chaque fois que le nombre de nœuds malveillants augmente. Mais, nous observons que le taux de suppression par l'approche IAACK est nettement plus faible en comparaison à l'approche AACK. Ceci peut être expliqué par le fait que l'approche IAACK détecte les nœuds malveillants et évite de faire acheminer les paquets de données à travers des routes impliquant ces nœuds. D'autre part, l'approche IAACK est capable de détecter et d'éviter les liens malveillants ce qui donne aux nœuds malveillants moins de chance de supprimer plus de paquets de données en s'impliquant dans de multiple routes.

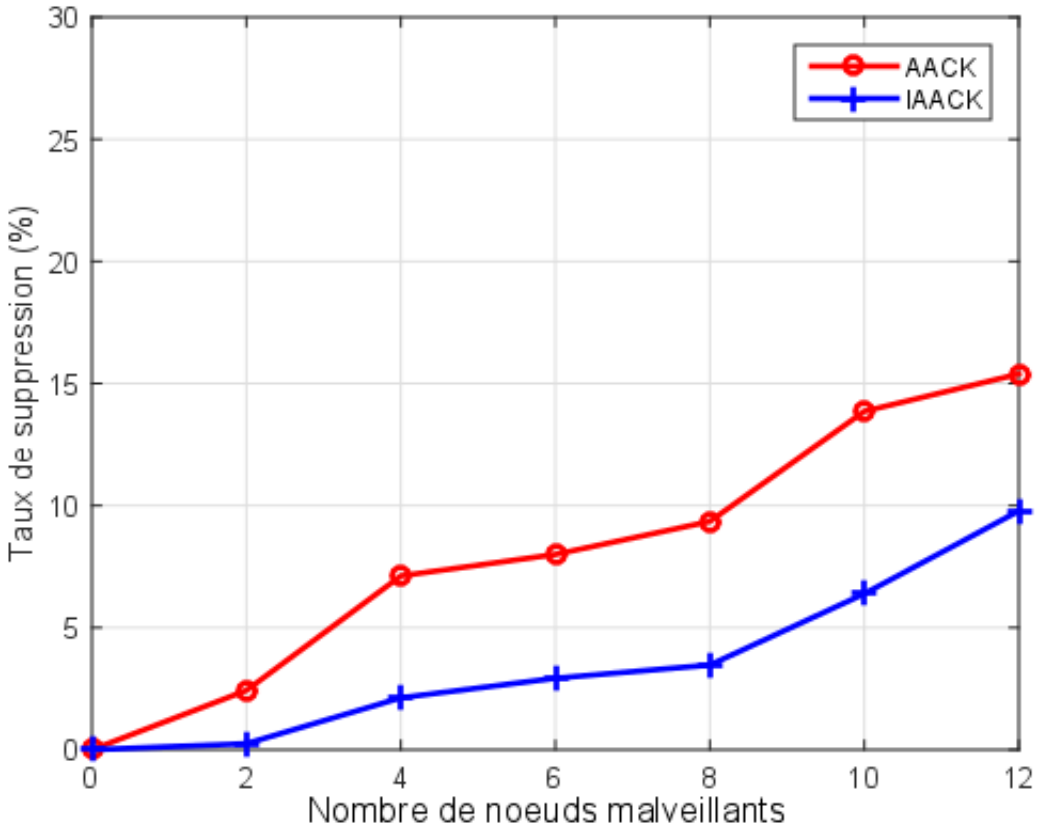


FIGURE 4.6 – Taux de suppression de paquets Vs nombre de nœuds malveillants (Vitesse = 10 m/s)

Afin d'illustrer l'impact de la vitesse des nœuds sur la performance des approches IAACK et AACK, la figure 4.7 et 4.8 présente le débit et le taux de suppression, respectivement. Dans ce cas, les figures sont tracées en fonction du nombre de nœuds malveillants et en fixant la vitesse des nœuds à 20 m/s. Conformément aux résultats présentés dans les figures 4.5 et 4.6, les résultats présentés dans les figures 4.7 et 4.8 démontrent que l'approche IAACK améliore le débit et réduit le taux de suppression des paquets de données en comparaison avec l'approche AACK (la différence devient plus apparente lorsque la vitesse des nœuds est fixée à 20 m/s). Ceci est dû au fait que : lorsque les nœuds se déplacent rapidement (vitesse élevée), leur voisinage change aussi (de nouveaux voisins). Étant donné que l'approche AACK n'exclut que les liens malveillants, chaque nouveau voisin constitue une chance pour construire un lien malveillant, et par conséquent supprimer des paquets données. L'approche IAACK est résistante à un tel changement de voisinage, car elle est capable d'éviter les nœuds malveillants dans le processus de sélection de routes.

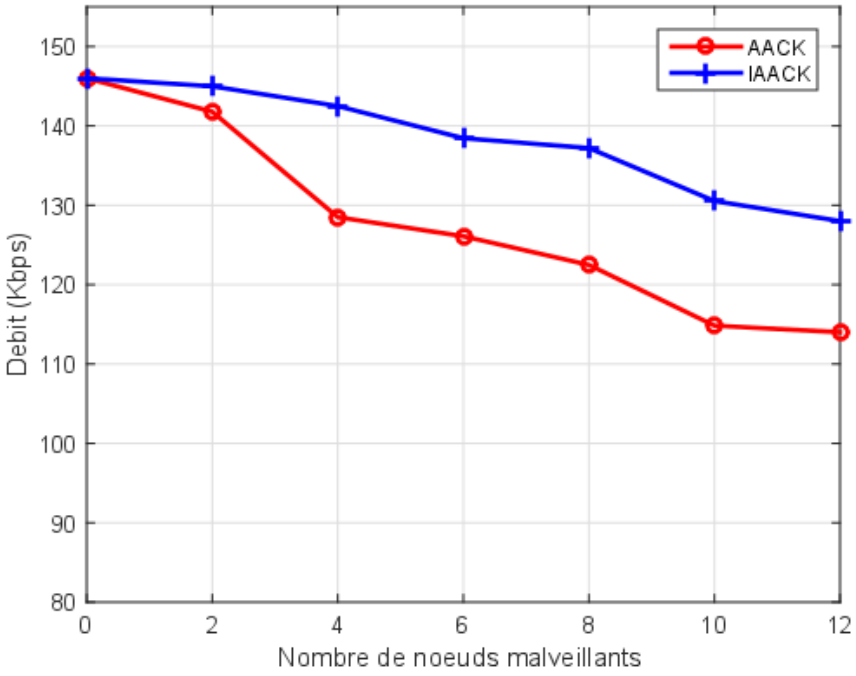


FIGURE 4.7 – Débit Vs nombre de nœuds malveillants (Vitesse = 20 m/s)

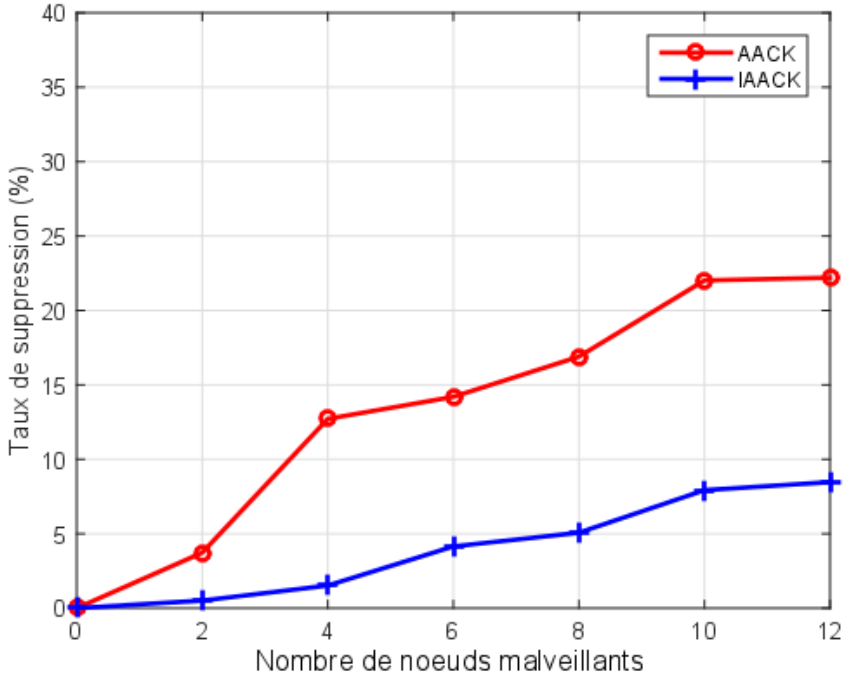


FIGURE 4.8 – Taux de suppression de paquets Vs nombre de nœuds malveillants (Vitesse = 20 m/s)

Les figure 4.9 et 4.10 présentent le taux de succès des nœuds égoïstes des approche IAACK et AACK. La vitesse des nœuds est fixée à 2 m/s et 20 m/s dans les figures 4.9 et 4.10, respectivement. A partir de la figure 4.9, nous pouvons observer que l'approche IAACK a un taux de succès égoïste faible en comparaison à l'approche AACK. Ceci parce que l'approche AACK n'incorpore pas un mécanisme d'incitation et de coopération des nœuds. Cependant, dans l'approche IAACK, un nœud a besoin de coopérer afin de gagner des crédits pour pouvoir faire acheminer ses paquets. Autrement, ses paquets seront rejetés. La différence en terme de taux de succès devient plus significatif lorsque la vitesse des nœuds est fixée à 20 m/s. Dans ce cas (la figure 4.10), on a un changement fréquent de routes (Envoi des paquets RREQ). Donc, les nœuds égoïstes refusant d'acheminer les paquets de routage des autres nœuds consomment leurs crédits rapidement, ce qui cause leur identification comme des nœuds égoïstes.

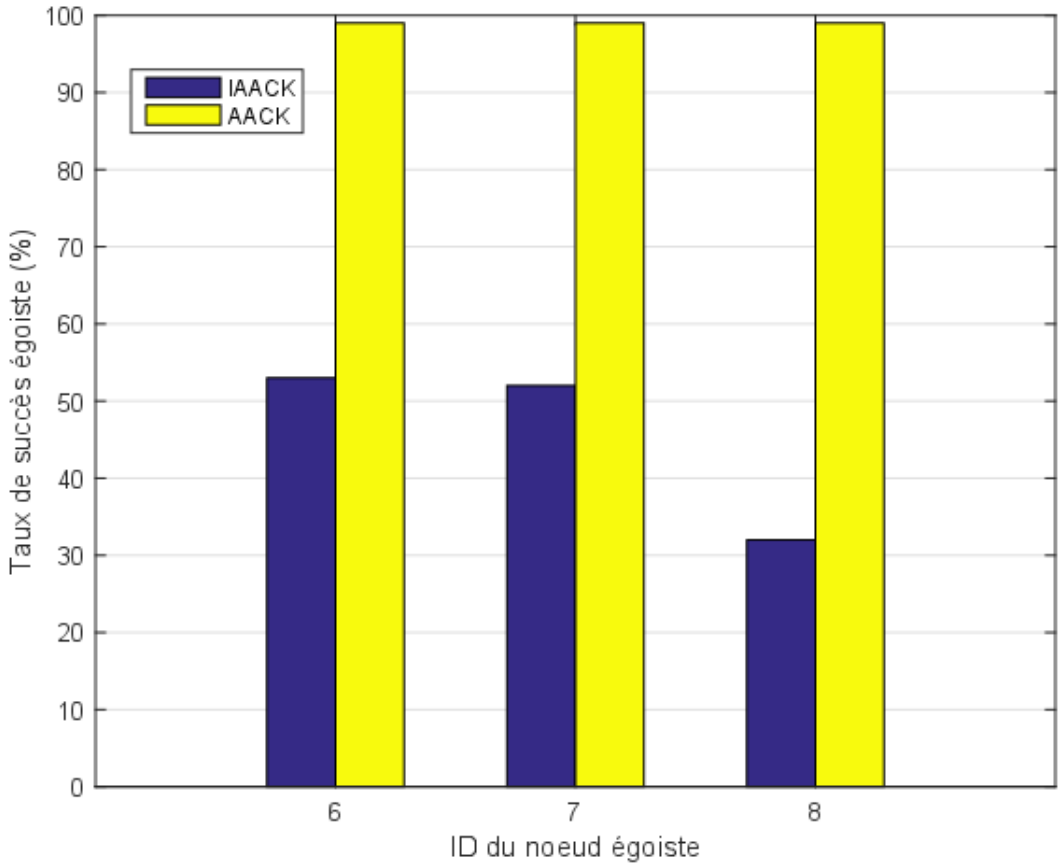


FIGURE 4.9 – Taux de succès égoïste (Vitesse = 2 m/s)

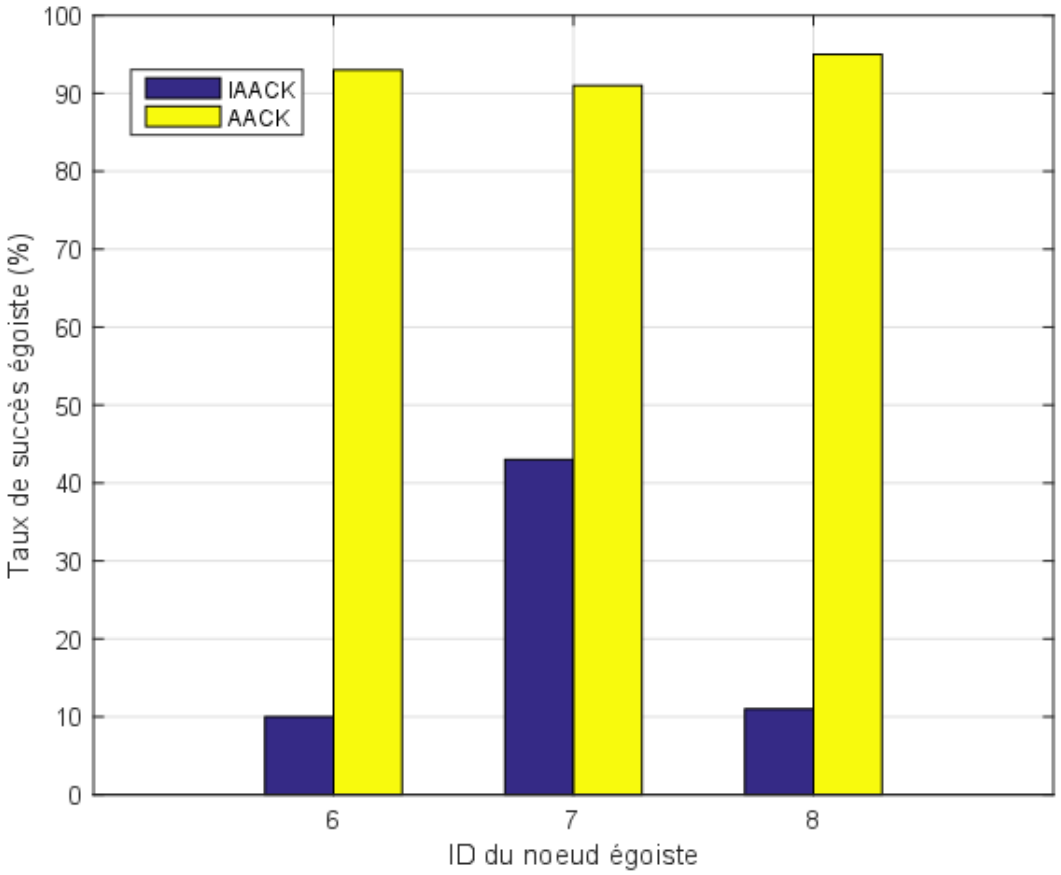


FIGURE 4.10 – Taux de succès égoïste (Vitesse = 20 m/s)

4.6 conclusion

Dans ce chapitre nous avons présenté les différents simulateurs de réseaux. Comme simulateur, notre choix est porté sur le simulateur NS version 2.34. Ensuite, les étapes necessaires qu'on a suivi pour implémenter notre approche IAACK y sont présentées. En utilisant le simulateur NS-2.34, nous avons conduit une série de simulations afin d'examiner la performance de notre approche en compraison avec l'approche AACK. Les résultats de simulation obtenus démontrent que l'approche IAACK permet d'améliorer le débit, réduire le taux de suppression des nœuds malveillants, tout en minimisant le taux de succès des nœuds égoïstes en comparaison avec l'approche AACK.

Conclusion et Perspectives

Dans ce mémoire, nous nous sommes concentrés sur le probléme de sécurité dans les réseaux mobiles ad hoc. Plusieurs protocoles de routage ont été proposés pour garantir la communication entre les nœuds. La plupart de ces protocoles repose sur la coopération de tous les nœuds pour bien fonctionner. Toutefois, une telle coopération ne peut pas être assurée pour cause des caractéristiques spécifiques d'un réseau mobile ad hoc. Dans notre mémoire, nous nous sommes focalisés sur l'attaque de suppression de paquets relative au protocole DSR.

L'état de l'art sur les approches proposées pour contrecarrer l'attaque de suppression de paquets nous a permis de cerner notre vision sur l'amélioration des approches à base d'acquittement. Nous avons constaté que ces approches souffrent de plusieurs limitations à savoir l'incapacité de : punir les nœuds malveillants sévèrement (détecter des liens malveillants au lieu des nœuds malveillant), détecter l'attaque collusion et stimuler la coopération des nœuds.

Pour remédier à ces limitations, nous avons proposé l'approche IAACK. Dans cette approche, nous avons modifié l'approche AACK afin de la rendre résistante à l'attaque collusion. Ainsi, nous avons proposé une méthode de calcul de réputation qui permet d'évaluer la réputation des nœuds au lieu de la réputation des liens. Comme moyen d'incitation, nous avons employé des crédits afin de rendre la coopération des nœuds dans le processus de routes bénéfique aux nœuds.

En utilisant le simulateur de réseau NS-2.34, nous avons évalué les performances de l'approche IAACK en comparaison avec l'approche AACK. Les résultats de simulation obtenus sont très favorables pour notre approche en termes de débit, taux de suppression de nœuds malveillants et taux de succès des nœuds égoïstes.

Notre objectif à court terme est d'étudier les performances de notre approche dans le cas de l'attaque de collusion en conduisant une série de simulation à l'aide du simulateur NS-2.34. Aussi, il est intéressant d'essayer d'adapter notre approche à un protocole de routage proactif tel que OLSR.

Bibliographie

[1] Gianni Di Caro, Frederick Ducatelle, and Luca Maria Gambardella. Anthocnet : an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, 16(5) :443–455, 2005.

[2] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. A comparative study of wireless protocols : Bluetooth, uwb, zigbee, and wi-fi. In *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, pages 46–51. Ieee, 2007.

[3] Angela Doufexi, Simon Armour, Michael Butler, Andrew Nix, David Bull, Joseph McGeehan, and Peter Karlsson. A comparison of the hiperlan/2 and ieee 802.11 a wireless lan standards. *IEEE Communications magazine*, 40(5) :172–180, 2002.

[4] Abderrezak Rachedi. *Contributions à la sécurité dans les réseaux mobiles ad Hoc*. PhD thesis, Université d'Avignon, 2008.

[5] Nadia Haddadou. *Réseaux ad hoc véhiculaires : vers une dissémination de données efficace, coopérative et fiable*. PhD thesis, Paris Est, 2014.

[6] J-C Cano and Pietro Manzoni. Evaluating the energy-consumption reduction in a manet by dynamically switching-off network interfaces. In *Computers and Communications, 2001. Proceedings. Sixth IEEE Symposium on*, pages 186–191. IEEE, 2001.

[7] Charles E Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *ACM SIGCOMM computer communication review*, volume 24, pages 234–244. ACM, 1994.

[8] Philippe Jacquet, Paul Mühlethaler, Thomas Clausen, Anis Laouiti, Amir Qayyum, and Laurent Viennot. Optimized link state routing protocol for ad hoc networks. In *Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings. IEEE International*, pages 62–68. IEEE, 2001.

[9] David B Johnson David A Maltz and Josh Broch. Dsr : The dynamic source routing protocol for multi-hop wireless ad hoc networks. *Computer Science Department Carnegie Mellon University Pittsburgh, PA*, pages 15213–3891, 2001.

[10] Charles Perkins, Elizabeth Belding-Royer, and Samir Das. Ad hoc on-demand distance vector (aodv) routing. Technical report, 2003.

[11] Zygmunt J Haas and Marc R Pearlman. The performance of query control schemes for the zone routing protocol. *IEEE/ACM Transactions on Networking (TON)*, 9(4) :427–438, 2001.

[12] Youcef Yahiatene. *Distribution de clés dans un réseau dynamique= Traffic encryption keys distribution models in mobile Ad hoc networks*. PhD thesis, 2012.

[13] Djamel Djenouri, Lyes Khelladi, and Nadjib Badache. A survey of security issues in mobile ad hoc networks. *IEEE communications surveys*, 7(4) :2–28, 2005.

[14] Mohammad Al-Shurman, Seong-Moo Yoo, and Seungjin Park. Black hole attack in mobile ad hoc networks. In *Proceedings of the 42nd annual Southeast regional conference*, pages 96–97. ACM, 2004.

[15] Rutvij H Jhaveri, Ashish D Patel, Jatin D Parmar, Bhavin I Shah, et al. Manet routing protocols and wormhole attack against aodv. *International Journal of Computer Science and Network Security*, 10(4) :12–18, 2010.

[16] Kashyap Balakrishnan, Jing Deng, and VK Varshney. Twoack : preventing selfishness in mobile ad hoc networks. In *Wireless communications and networking conference, 2005 IEEE*, volume 4, pages 2137–2142. IEEE, 2005.

[17] William Stallings. *Cryptography and network security : principles and practices*. Pearson Education India, 2006.

[18] Carlisle Adams and Steve Lloyd. *Understanding PKI : concepts, standards, and deployment considerations*. Addison-Wesley Professional, 2003.

[19] Dipali D Punwatkar and Kapil N Hande. A review of malicious node detection in mobile ad-hoc networks. *International Journal of Computer Sciences & Engineering (IJCSE)*, 2(2) :65–69, 2014.

[20] Seung Yi, Prasad Naldurg, and Robin Kravets. Security-aware ad hoc routing for wireless networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 299–302. ACM, 2001.

[21] Levente Buttyan and Jean-Pierre Hubaux. Nuglets : a virtual currency to stimulate cooperation in self-organized mobile ad hoc networks. Technical report, 2001.

[22] Sheng Zhong, Jiang Chen, and Yang Richard Yang. Sprite : A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1987–1997. IEEE, 2003.

[23] Sorav Bansal and Mary Baker. Observation-based cooperation enforcement in ad hoc networks. *arXiv preprint cs/0307012*, 2003.

[24] Jianli Guo, Hongwei Liu, Jian Dong, and Xiaozong Yang. Head : A hybrid mechanism to enforce node cooperation in mobile ad hoc networks. *Tsinghua Science & Technology*, 12 :202–207, 2007.

[25] Lei Huang, Li Lei, Liu Lixiang, Zhang Haibin, and Linsha Tang. Stimulating cooperation in route discovery of ad hoc networks. In *Proceedings of the 3rd ACM workshop on QoS and security for wireless and mobile networks*, pages 39–46. ACM, 2007.

[26] Mahdi Bounouni and Louiza Bouallouche-Medjkoune. A hybrid stimulation approach for coping against the malevolence and selfishness in mobile ad hoc network. *Wireless Personal Communications*, 88(2) :255–281, 2016.

[27] Sergio Marti, Thomas J Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265. ACM, 2000.

[28] Sonja Buchegger and Jean-Yves Le Boudec. Performance analysis of the confidant protocol. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 226–236. ACM, 2002.

[29] Pietro Michiardi and Refik Molva. Core : a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Advanced communications and multimedia security*, pages 107–121. Springer, 2002.

[30] Kejun Liu, Jing Deng, Pramod K Varshney, and Kashyap Balakrishnan. An acknowledgment-based approach for the detection of routing misbehavior in manets. *IEEE transactions on mobile computing*, 6(5), 2007.

[31] Tarek Sheltami, Anas Al-Roubaiey, Elhadi Shakshuki, and Ashraf Mahmoud. Video transmission enhancement in presence of misbehaving nodes in manets. *Multimedia systems*, 15(5) :273–282, 2009.

[32] Elhadi M Shakshuki, Nan Kang, and Tarek R Sheltami. Eaackâ"a secure intrusion-detection system for manets. *IEEE Transactions on industrial electronics*, 60(3) :1089–1098, 2013.

[33] Matthew Sexton, Edward Smith, and Bernie Eydt. Wireless security. *Wiley Handbook of Science and Technology for Homeland Security*, 2009.

[34] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2) :120–126, 1978.

[35] Kevin Fall and Kannan Varadhan. The ns manual (formerly ns notes and documentation). *The VINT project*, 47, 2005.

[36] Jorge Nuevo. A comprehensible glomosim tutorial. *Université du Quebec March*, 4, 2004.

[37] Xiang Zeng, Rajive Bagrodia, and Mario Gerla. Glomosim : a library for parallel simulation of large-scale wireless networks. In *Parallel and Distributed Simulation, 1998. PADS 98. Proceedings. Twelfth Workshop on*, pages 154–161. IEEE, 1998.

[38] András Varga and Rudolf Hornig. An overview of the omnet++ simulation environment. In *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, page 60. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[39] Ahmed Sobeih, Wei-Peng Chen, Jennifer C Hou, Lu-Chuan Kung, Ning Li, Hyuk Lim, Hung-Ying Tyan, and Honghai Zhang. J-sim : A simulation environment for wireless sensor networks. In *Proceedings of the 38th annual Symposium on Simulation*, pages 175–187. IEEE Computer Society, 2005.

[40] Teerawat Issariyakul and Ekram Hossain. Introduction to network simulator ns2 manual, 2009.

Annexe

La méthode d'installation de NS2 :

L'installation du simulateur NS2 se fera en quatre étapes que nous allons présenter ci-dessous :

Etape 1 :

- Après l'installation de NS2.34 nous l'avons enregistré dans le répertoire « /home/elhadi /Documents ».
- Nous avons mis-à jour UBUNTU on exécutant les commandes suivantes :

– **sudo apt-get update :** L'option update met à jour la liste des fichiers disponibles dans les dépôts APT présents dans le fichier de configuration /etc./apt/sources.list.

– **sudo apt-get dist-upgrade :** L'option dist-upgrade met à jour tous les paquets installés vers les dernières versions en installant de nouveaux paquets si nécessaire.

– **sudo apt-get upgrade :** L'option upgrade met à jour tous les paquets installés sur le système vers les dernières versions .

- Avant d'installer NS, nous avons installé certains packages essentiels requis pour NS2 qui sont :

– **Sudo apt-get install build-essential autoconf automake :**Permet d'installer le logiciel de base dont on avez besoin pour compiler à partir des sources, comme le compilateur GCC et d'autres services.

– **Sudo apt-get install tcl8.5-dev tk8.5-dev** Package de TCL.

– **Sudo apt-get install perlxgraphlibxt-dev libx11-dev libxmu-dev :**Package du NAM

– **Sudo apt-get install gcc-4.4 :**Permet d'installer le GNU MAKE afin de pouvoir compiler le logiciel en C / C ++.

Etape 2 : Extraire et installer NS2

- Nous avons accédé au dossier où NS2 à été installé et nous l'avons extrait. On utilisant ces deux commandes à partir du terminal :

1	CD/home/elhadi/Documents
2	Tar-xvzf ns-allinone-2.34.tar.gz

- Après avoir extrait le dossier "ns-allinone-2.34", nous avons modifié le fichier "/ns-allinone-2.34/ns-2.34/linkstate/ls.h" dans la ligne 137 en remplaçant le mot "erase" par "this⇒erase≫.
- En suite nous avons commencé l'installation après l'accès au dossier décompressé du NS2. En tapant ces deux commandes :

1	CD/home/elhadi/Documents/ns-allinone-2.34
2	Sudo ./install

Etape 3 : La définition des variables d'environnements

Après avoir installé NS correctement.Nous avons rajouté des variables d'environnement au niveau du fichier bashrc en utilisant la commande suivante :

1	Sudo gedit .bashrc

- Ensuite nous avons ajouté le code afficher ci-dessous à la fin du fichier bashrc en remplaçant ≪ /path_ to ≫ par le chemin dont lequel nous avons mis le dossier du ns2, ≪ /home/elhadi/documents ≫.

LD_ LIBRARY_ PATH
OTCL_ LIB=/home/elhadi/documents/ns-allinone-2.34/otcl-1.14/
NS2_ LIB=/home/elhadi/documents/ns-allinone-2.34/lib/
USR_ Local_ LIB=/usr/local/lib/
export LD_ LIBRARY_ PATH=$ LD_ LIBRARY_ PATH :$OTCL_ LIB :$NS2_ LIB :$USR_ Local_ LIB
TCL_ LIBRARY
TCL_ LIB=/home/elhadi/documents/ns-allinone-2.34/tcl8.5.10/library/
USR_ LIB=/usr/lib/
export TCL_ LIBRARY=$TCL_ LIBRARY :$TCL_ LIB :$USR_ LIB
PATH
XGRAPH=/home/elhadi/documents/ns-allinone-2.34/xgraph-12.2/ :/home/elhadi/documents/ns-allinone-2.34/bin/ :/home/elhadi/documents/ns-allinone-2.34/tcl8.5.10/unix/ :/home/elhadi/documents/ns-allinone-2.34/tk8.5.10/unix/
NS=/home/elhadi/documents/ns-allinone-2.34/ns-2.34/
NAM=/home/elhadi/documents/ns-allinone-2.34/nam-1.15/
export PATH=$ PATH :$ XGRAPH :$NS :$ NAM

- Par la suite, nous avons redémarré le système pour qu'il prenne en compte les modifications accordés et nous avons rechargé le fichier bashrc en utilsant la commande suivante :

```
1 source ~/.bashrc
```

Etape 4 : Vérification de l'installation

La dernière étape consiste a la vérification du bon fonctionnement de NS2. Pour cela, nous avons utilisé les deux commandes suivantes :

```
1 CD/home/elhadi/Documents/ns-allinone-2.34/ns-2.34
2 ./validate
```

Résume

Un réseau mobile ad hoc (MANET) est une collection de nœuds capables de communiquer sans l'aide d'une infrastructure préexistante ou administration centralisée. Plusieurs protocoles de routage ont été proposés pour garantir la communication entre les nœuds du réseau. Ils se basent sur l'hypothèse que tous les nœuds sont disposés à coopérer. Cependant, une telle coopération ne peut être garantie pour cause de leur caractéristiques spécifiques, qui les rendent vulnérables à plusieurs attaques. Dans ce mémoire, nous nous sommes focalisés sur l'attaque de suppression de paquets. Nous avons constaté que les approches d'acquittements proposées pour contrecarrer l'attaque de suppression de paquets souffrent de plusieurs limitations. Afin de remédier à ces limitations, nous avons proposé une approche appelé IAACK (Improved AACK). L'approche IAACK est structurée autour de trois composants. Le composant de surveillance est responsable de surveiller l'acheminement correcte des paquets de données et de détecter des éventuelles collusions entre les nœuds malveillants. Le composant de réputation est responsable de l'évaluation des valeurs de réputation des nœuds et de l'isolation des nœuds malveillants. Le composant de stimulation a comme fonction la stimulation de la coopération des nœuds dans le processus de découverte de routes en utilisant des crédits comme incitation. Les résultats de simulation démontrent que l'approche IAACK permet d'améliorer le débit, tout en réduisant le taux de suppression des nœuds malveillants et le taux de succès des nœuds égoïstes.

Mots clés : Réseaux mobile Ad Hoc, nœud malveillant, nœud égoïste, attaque de suppression de paquets, IAACK, NS-2.34

Abstract

A mobile ad hoc network (MANET) is a collection of nodes that are able to communicate without the help of a pre-existing infrastructure or centralized administration. Several routing protocols have been proposed to ensure communication between nodes. Almost of them rely on the assumption that all nodes are willing to cooperate. However, such cooperation can not be guaranteed. The specifics characteristics of MANET make them vulnerable to several attacks. In this document, we focused on the packet dropping attack. We have noticed that the acknowledgment approaches proposed to cope against packet dropping attack suffer from several limitations. In order to cope these limitations, we have proposed IAACK approach. IAACK approach is organized around three components.The monitoring component is responsible for monitoring the correct forwarding of data packets and for detecting eventual collision between malicious nodes. The reputation component is responsible for the evaluation of nodes reputation values and the isolation of malicious nodes. The stimulation component is responsible for motivating the cooperation of nodes in the route discovery process using credits as incentive. The simulation results demonstrate that our approach IAACK improves the throughput, while at the same time reduces the malicious dropping ratio and the success rate of selfish nodes.

Key words : Mobile Ad Hoc networks, malicious node, Selfish node, packet dropping attack, IAACK, NS-2.34